Fluid approximation of broadcasting systems

Fluid approximation of broadcasting systems[1]

Luca Bortolussi[a], Jane Hillston[b], Michele Loreti[c]

[a] *Dipartimento di Matematica e Geoscienze, Università di Trieste*
[b] *Laboratory for Foundations of Computer Science, University of Edinburgh*
[c] *Scuola di Scienze e Tecnologie, Università di Camerino*

Abstract

Nature-inspired paradigms have been proposed to design and forecast behaviour of open distributed systems, such as sensor networks and the internet of things. In these paradigms system behaviour emerges from (complex) interactions among a large number of agents. Modelling these interactions in terms of classical point-to-point communication is often not practical. This is due to the large scale and the open nature of the systems, which means that partners for point-to-point communication may not be available at any given time. Nevertheless the need for efficient formal verification of qualitative and quantitative properties of these systems is of utmost importance, especially given their proposed pervasive and transparent nature.

CARMA is a recently proposed formal modelling language for open distributed systems, which is equipped with a broadcast communication in order to meet the communication challenges of such systems. The inclusion of quantitative information about the timing and probability of actions gives rise to models suitable for analysing questions such as the probability that information will achieve total coverage within a system, or the expected market share that might be gained by competing service providers relying on viral advertising. The ability to express models is not the only challenge, because the scale of the systems we are interested in often defies discrete state-based analysis techniques such as stochastic simulation. This is the problem that we address in this paper as we consider how to provide an efficient fluid approximation, supporting efficient and accurate quantitative analysis of large scale systems, for a language that incorporates broadcast communication.

Keywords: Natural inspired paradigms, Broadcast Communication, Stochastic Process Algebra, Fluid Approximation, Population Models, Open Distributed Systems.

1. Introduction

As the digital world becomes ever more connected, systems are increasingly developed to comprise of large numbers of interacting agents. Often in these systems the behaviours of the individual agents are relatively simple but the complex capabilities of the system are derived from the interactions and the sheer number of agents involved. Examples include wireless sensor networks such as those used to monitor

and respond to fires within buildings [28] and Internet of Things applications such as smart energy networks where appliances are remotely switched on and off in order to maintain supply-demand balance within the network as a whole [35].

Nature-inspired paradigms [23] have been proposed to design and forecast behaviour of open distributed systems. Following these paradigms, system behaviour emerges from (complex) interactions among participating agents that, in turn, are influenced by the state of the overall system. An example is *swarm intelligence* [21] that is heavily applied to optimisation problems in many fields.

In such systems the communication patterns between the agents are crucial for achieving the desired functionality. Pairwise point-to-point communication proves to be inadequate in these systems for two key reasons. Firstly, the open nature of the systems means that agents may be unresponsive because they are disconnected or switched off, meaning that the participants in an interaction cannot be known ahead of time. This motivates the need for a non-blocking communication pattern, which allows an agent to proceed even when it cannot find a communication partner. Secondly, the large size of the system in terms of number of agents will mean that pairwise communication will simply be too slow when information needs to be disseminated to all participants. This motivates the need for a broadcast or one-to-many communication pattern.

In order to properly describe the behaviour of these systems, it is useful to have a formal specification language that can also be used to reason about the possible behaviours. Specifically, in this paper, we focus on the family of stochastic process algebras such as PEPA [26], EMPA [5] and the Stochastic Pi-Calculus [31], assuming that an exponential distribution is associated with each action, giving it a randomly distributed duration, and leading to an underlying semantics in terms of a Continuous Time Markov Chain (CTMC). CTMCs support quantitative analysis of models to predict how systems will behave over time, and answer questions about timeliness of responses, expected throughput and resource usage. Thus they provide a valuable means of assessing the dynamic behaviour of systems before they are deployed, checking that both user and system operator non-functional requirements will be met. However, the CTMCs capturing the type of large open distributed systems that we consider in this paper are often so large that standard CTMC analysis techniques, based on the discrete state representation, become highly inefficient or even intractable.

In recent years techniques have been developed to map stochastic process algebra models to efficient fluid approximations. The underlying mathematics assumes that as the system size grows the impact of individual actions diminishes. The existing fluid approximation results have been developed for languages in which communication is synchronous and unicast. The challenge in the context of open distributed systems is to combine this approach to CTMC analysis with models that facilitate broadcast communication. The difficulty lies in the fact that a broadcast communication has the potential to change the state of all agents in the system at once, violating the usual assumption that the impact of any action is diminished as the number of agents grows.

The starting point of this paper is thus a stochastic process algebra supporting broadcast communication, specifically CARMA [12]. CARMA is an expressive language, supporting both point-to-point and a broadcast-based communication, using attributes to identify the subset of potential receivers. In a CARMA model a collective of agents operate in the context of an explicit environment, which shapes the way in

which agents interact. Currently CARMA practitioners rely on a tool to develop their models and analyse their performance via stochastic simulation [27]. However, such simulations can be computationally intensive and become a bottleneck for the analysis of large models, particularly when large populations of agents are involved. Therefore there is a strong motivation for developing a fluid approximation of CARMA models as a viable alternative. However, as explained above, whilst fluid semantics, leading to fluid approximations, have been defined for process algebras like PEPA with synchronous communication [25] over a decade ago, the definition of a fluid semantics for a language supporting broadcast communication is much more challenging. The convergence results which justify the mean field approximation rely on each action inducing a bounded amount of change in the population. Broadcast by its nature has the potential to induce change in every receiving agent in the system. Thus the notions of broadcast and fluid approximation appear to be at odds.

In order to tackle this problem, the definition of broadcast in CARMA must be carefully constructed, in order to provide a balance between a useful representation that supports the expression of realistic models, but which nevertheless admits a sound fluid approximation supporting efficient quantitative analysis.

Paper contribution. The main contribution of this paper is the identification of proper restrictions on CARMA that guarantee that we can define a sound fluid approximation. To start with, we strip CARMA of features which are not relevant to this paper. In particular, we limit ourselves to a fragment of CARMA containing unicast and broadcast communication primitives, but without attributes. Intuitively, this corresponds to the assumption that attributes have been incorporated into the agent states, possibly exploding their number. We then provide this fragment of CARMA, that we named NBA (Network of Broadcasting Agents), with a population based semantics, in which the focus is switched from the individual agents, to the proportion of a population that is behaving in the same way. Based on this population semantics, we formally derive the fluid equations, and study under which conditions the convergence results hold. This enables us to identify appropriate scaling laws for the broadcast, which essentially require that the number of agents receiving a message remains constant on average, independently of the population size. This restriction has a natural interpretation, as each communication channel has physical constraints limiting its actual capacity. We support this construction with a convergence theorem, proving the correctness of our approach.

The mean field approximation is demonstrated on examples representing collective adaptive system applications, such as a *collaborative agent scenario* (which is used as a running example) and *information sharing through gossip protocols*.

Related Work. In open distributed systems, such as swarms of robots, the broadcast communication paradigm is more appropriate than unicast [15]. Thus when we develop formal methods to allow us to reason about the behaviour of such systems, our formalisms must also incorporate broadcast communication as a primitive. Similarly the open nature of the systems, with a changing population of agents, favours non-blocking communication. There have been a variety of formalisms proposed for capturing the behaviour of populations of agents [20, 33, 16] but most of these are focused

on synchronous communication. In contrast, the early work by Prasad developed the Calculus of Broadcasting Systems (CBS) [30], a qualitative calculus where individual agents interact via broadcast. Actions in the calculus were given a priority, used to choose which action would fire first. The broadcast primitives within CARMA are inspired by CBS but incorporate timing information in order to support quantitative verification of systems. More recent broadcast-based calculi rely on attribute based communication to identify a subset of agents to engage with messages. These include SCEL [20] and ABC [1], which have a qualitative semantics, and PALOMA [22] and CARMA [12] from the family of stochastic process calculi. In these two calculi, communication happens via broadcast (and synchronous message passing) and agents can receive a message with a certain probability. In PALOMA there is a mechanism to obtain differential equations for the first and second order moments of the underlying Markov process, but there is no consistent fluid semantics supporting a proof of convergence between the obtained ordinary differential equations and the underlying Markov process. Similarly, in the work of Bruneo *et al.* [14], which works more directly at the level of the underlying Markov process, a set of ordinary differential equations representing the population density of different agent classes within a model is systematically derived but not related to the original discrete state model.

Fluid approximation [18] is a powerful technique to analyse stochastic population models of interacting agents, well established in fields like performance modelling and systems biology [10]. It has been lifted to stochastic process algebra models [16, 25, 13], and extended in different directions, to deal with more complex scenarios, like different population scales, leading to hybrid limits [6, 4], uncertainty in model parameters [7], spatio-temporal models [34]. It has also been used to speed up verification of formal properties of population models [9, 11] and to compute rewards [8].

Paper structure. In Section 2 we introduce NBA, a simple quantitative calculus for broadcasting processes. Section 3, instead, introduces Markov population processes, which will be used to express the population semantics of NBA. There we also introduce a version of the fluid approximation theorem suitable for our purposes. In Section 4 we show how to obtain a population semantics for NBA, while Section 5 is devoted to the discussion of a case study of a Gossip Shuffle Protocol. Final remarks are drawn in Section 6.

2. Modelling quantitative aspects of broadcasting systems

In this section we present NBA, *Network of Broadcasting Agents*, a fragment of CARMA without attributes, but still supporting unicast and broadcast communication, that can be used to model quantitative aspects of systems of broadcasting processes. The syntax of this calculus as presented here is inspired by CBS, the Calculus of Broadcasting Systems introduced in [30].

Example 1. To present NBA we consider a simple scenario. We have a group of agents that can be either *red* or *blue*. We want to guarantee that the two groups are *balanced* in size — without any centralised control. Each agent can choose/change its colour only by observing the interactions with the other participants in the systems. To guarantee

the equilibrium between the two kind of agents the following interaction schema is used. All agents *publish* their colour. A *transitional* phase is started when an agent *meets* someone of the same colour. In this situation an agent asks another agent with the same colour to *change*. The *transitional* phase can be cancelled when an agent with a different colour is found. As typical of this kind of system, the procedure never ends. □

A system modelled with NBA consists of a *multiset* of interacting agents, each of which is identified by a name in AG. This is a set of *agent identifiers* whose elements are denoted by $A, A_1, A', \ldots, B, B_1, B', \ldots$.

Example 2. To model the *scenario* of Example 1 we consider four agents R, B, RT, BT, where R and B identifies *red* and *blue* agents, respectively, while RT and BT identifies the same coloured agents in a *transitional* phase.

□

We let SYS be the set of *systems* S generated by the following syntax:

$$S \quad ::= \quad \mathbf{0} \;\Big|\; A \in \text{AG} \;\Big|\; S_1 \parallel S_2 \tag{1}$$

Each system S can be also thought of as a *multisets* of agents. Under this perspective, each term S can be interpreted as a function mapping each agent identifier A to an integer k (denoted by $S[A]$) indicating the number of occurrences of A in the system.

Basic systems can be either $\mathbf{0}$ or A. The former denotes the empty system, i.e. the one where no agent is running. The latter represents the system where a single instance of agent A exists. Systems are composed via operator $\parallel$: $S_1 \parallel S_2$ denotes the system where all the agents in S_1 and in S_2 coexist. In the rest of this paper we will use $k \cdot S$ to denote the system consisting of k copies of S:

$$k \cdot S = \underbrace{S \parallel \cdots \parallel S}_{n \text{ times}}$$

We can observe that $k \cdot S$ corresponds to the function associating $k \cdot S[A]$ to each A. Moreover, given a system $S \in \text{SYS}$ and an agent $A \in \text{AG}$, we let $S - A$ denotes the system obtained from S by removing one instance of A. We will also use $A[k]$ to denote $k \cdot A$.

Example 3. Let us consider the system of Example 3. A general configuration of our system has the form:

$$\mathsf{R}[k_1] \parallel \mathsf{B}[k_2] \parallel \mathsf{RT}[k_3] \parallel \mathsf{BT}[k_4]$$

where $k_1, k_2, k_3, k_4 \in \mathbb{N}$ indicate the number of agents in the different states we have in the system. These values can be parametrised with respect to an integer value N indicating the *scale* of our system. If we assume that at the beginning we have $97 \cdot N$ agents in state R, and N instances of agents RT, B and BT, the initial configuration can be expressed as:

$$\mathsf{System} \triangleq \mathsf{R}[97 \cdot N] \parallel \mathsf{B}[N] \parallel \mathsf{RT}[N] \parallel \mathsf{BT}[N]$$

□

In NBA each agent is associated with a *behaviour*. The latter is described via a *process term P* in the set PROC. We let ADEF denote the set of *agents definition* Δ, that is functions in AG $\to$ PROC. We also let PROC be the set of *process terms* denoted by $P, P_1, P', \ldots, Q, Q_1, Q' \ldots$ defined by the following syntax:

$$\begin{array}{rcl} P & ::= & \mathbf{nil} \mid \alpha.A \mid P_1 \oplus_p P_2 \\ \alpha & ::= & a? \mid a! \mid a^\star? \mid a^\star! \end{array} \tag{2}$$

where p is a real value in $[0,1]$, α is an *action* in the set ACT while a is a *channel* in CH.

The syntax of elements in PROC is similar to many probabilistic process algebras already defined in literature. The term **nil** denotes the *inactive agent*, while $\alpha.A$ represents the process that first executes action α and then becomes A. Actions α are used to perform interactions with other agents.

In our calculus two kind of interactions can occur: *unicast interaction* and *broadcast interaction*. The first kind of synchronisation represents a *one-to-one* interaction: one agent sends a message, by executing a *unicast output* ($a!$), that is received by an agent executing the complementary *unicast input* ($a?$). *Broadcast interactions* are used to model *one-to-many* communications: one agent sends a message via a *broadcast output* ($a^\star!$) that can be received by the agents that are executing the complementary *broadcast input* ($a^\star?$). Notice that while in the *unicast synchronisation* both input and output are *blocking*, in the *broadcast synchronisation* an output can be performed even if there are no agents ready to receive the message.

Different behaviours can be combined by using the probabilistic choice operator $P_1 \oplus_p P_2$. Intuitively, this indicates that an action is executed by P_1 with probability p and by P_2 with probability $1-p$. We assume $p = 0.5$ when it is omitted, namely $P_1 \oplus P_2$ stands for $P_1 \oplus_{0.5} P_2$; and similarly for choices with more elements. We will also use $\alpha.(A_1 \oplus_{p_1} \cdots \oplus_{p_{n-1}} A_n)$ to denote $(\alpha.A_1) \oplus_{p_1} \cdots \oplus_{p_{n-1}} (\alpha.A_n)$.

Example 4. We are now ready to define the behaviour of agents introduced in Example 2. We can observe that NBA communication primitives can be used to model the interactions informally described in Example 1. Indeed, *broadcast* is used to *publish* agent colour ($\mathsf{red}^\star!$ and $\mathsf{blue}^\star!$), while *unicast* triggers the switch from one colour to the other ($\mathsf{beBlue}!$ and $\mathsf{beRed}!$). Agents are defined as follows:

$$\begin{array}{rcl} \mathsf{R} & \triangleq & \mathsf{red}^\star!.\mathsf{R} \oplus \mathsf{red}^\star?.(\mathsf{RT} \oplus_{p_t} \mathsf{R}) \oplus \mathsf{beBlue}?.\mathsf{B} \\ \mathsf{B} & \triangleq & \mathsf{blue}^\star!.\mathsf{R} \oplus \mathsf{blue}^\star?.(\mathsf{BT} \oplus_{p_t} \mathsf{B}) \oplus \mathsf{beRed}?.\mathsf{R} \\ \mathsf{RT} & \triangleq & \mathsf{red}^\star!.\mathsf{RT} \oplus \mathsf{beBlue}!.\mathsf{R} \oplus \mathsf{beBlue}?.\mathsf{B} \oplus \mathsf{blue}^\star?.(\mathsf{R} \oplus_{p_c} \mathsf{RT}) \\ \mathsf{BT} & \triangleq & \mathsf{blue}^\star!.\mathsf{BT} \oplus \mathsf{beRed}!.\mathsf{B} \oplus \mathsf{beRed}?.\mathsf{R} \oplus \mathsf{red}^\star?.(\mathsf{B} \oplus_{p_c} \mathsf{BT}) \end{array}$$

Agent R uses *broadcast output* to *advertise* its colour ($\mathsf{red}^\star!.\mathsf{R}$). Another instance of R that receives this message enters in a transitional phase with probability p_t while ignores the message with probability $1-p_t$ ($\mathsf{red}^\star?.(\mathsf{RT} \oplus_{p_t} \mathsf{R})$). Finally, R can receive a beBlue message to change its colour and become B ($\mathsf{beBlue}?.\mathsf{B}$). Agent RT identifies a red agent that is in a *transitional* phase. Like an agent R, RT can advertise its colour to other agents ($\mathsf{red}^\star!.\mathsf{RT}$). Moreover, it executes action beBlue! to trigger the change. The transitional phase is completed when the action beBlue? is executed and RT becomes B. Finally, the transitional phase can be *cancelled* when RT executes action $\mathsf{blue}^\star?$. In this case RT evolves with probability p_c to R and with probability $1-p_c$ it remains in the same state.

Agents B and BT are similar, but for the fact that *red* and *blue* are inverted.

□

The behaviour P associated with an agent A describes the possible interactions an agent can perform. However, P does not provide any explicit information about the *quantitative aspects* of these interactions. In NBA, like in many other stochastic process algebras, we assume that the execution of an action/interaction takes time. This duration is governed by a random variable that is *exponentially distributed*. Hence, to compute the *duration* of a synchronisation we have to associate a *rate* with each action an agent can perform. Moreover, we have also to provide the appropriate mechanisms to compute the probability distribution used to select the receiver of a *unicast* or *broadcast* output. Note that all this *quantitative information* may depend on the number of agents that are currently activated in the system.

To model all these aspects, we use an *environment* $\varepsilon \in \mathsf{Env}$. This is a function that, given a system S, yields a tuple $\langle \mathbf{r}, \mathbf{p}, \mathbf{w} \rangle$ of functions that are used to compute rates of output actions and the probability that a given agent receives a message:

- $\mathbf{r} : \text{AG} \times \text{ACT} \to \mathbb{R}_{\geq 0}$ computes the execution rate of action α executed by an agent A;
- $\mathbf{p} : \text{AG} \times \text{ACT} \to [0,1]$ expresses the probability that an agent A receives a broadcast message;
- $\mathbf{w} : \text{AG} \times \text{ACT} \to \mathbb{R}_{\geq 0}$ yields the weight that will be used to compute the probability that an agent A can receive a unicast message.

Let ε be an *environment*, and $S \in \text{SYS}$, we will use $\mathbf{r}^\varepsilon_S$, $\mathbf{p}^\varepsilon_S$ and $\mathbf{r}^\varepsilon_S$ whenever $\varepsilon(S) = \langle \mathbf{r}^\varepsilon_S, \mathbf{p}^\varepsilon_S, \mathbf{w}^\varepsilon_S \rangle$. Moreover, for any agent $A \in \text{AG}$ we let $\mathbf{E}^S_\varepsilon[A]$ denote *exit rate*, that is the total rate of the actions A can perform: $\mathbf{E}^S_\varepsilon[A] = \sum_{\alpha \in \text{ACT}} \mathbf{r}^\varepsilon_S(A, \alpha)$.

Example 5. To perform quantitative analysis of the model of Example 4 we have to define an *environment* ε_{ca}. As we have defined above, this function associates each system S with the tuple of functions $\langle \mathbf{r}^{\varepsilon_{ca}}_S, \mathbf{p}^{\varepsilon_{ca}}_S, \mathbf{w}^{\varepsilon_{ca}}_S \rangle$.

Function $\mathbf{r}^{\varepsilon_{ca}}_S$ associates a *rate* with each (output) action that an agent can perform. These actions are associated to the *events* that can occur in a system. In our case we have: color advertisement (actions $\mathsf{red}^\star!$ and $\mathsf{blue}^\star!$), and the execution of an action triggering the colour change (actions beBlue! and beRed!). Rates of the above actions are computed by function $\mathbf{r}^{\varepsilon_{ca}}_S$ that is defined as follows:

- $\mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{R},\mathsf{red}^\star!) = \mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{RT},\mathsf{red}^\star!) = \mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{B},\mathsf{blue}^\star!) = \mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{BT},\mathsf{blue}^\star!) = \lambda_a$;
- $\mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{RT},\mathsf{beBlue}!) = \mathbf{r}_S^{\varepsilon_{ca}}(\mathsf{BT},\mathsf{beRed}!) = \lambda_c$.

Above λ_a is the *advertising rate*, while λ_c indicates the rate at which an agent can *change* its colour.

Function $\mathbf{p}_S^{\varepsilon_{ca}}$ identifies the probability that an agent receives a *broadcast message*. Here we assume that each *advertisement* can be received on average by $k_r \in \mathbb{N}_{>0}$ agents. This value does not depend on the number of participants in the system, while it is related to its communication capabilities, i.e. the capacity of a message to reach other agents. Function $\mathbf{p}_S^{\varepsilon_{ca}}$ is the following:

$$\begin{aligned}
\mathbf{p}_S^{\varepsilon_{ca}}(\mathsf{R},\mathsf{red}^\star?) &= \mathbf{p}_S^{\varepsilon_{ca}}(\mathsf{BT},\mathsf{red}^\star?) &= \min\left\{1, \frac{k_r}{S[\mathsf{R}]+S[\mathsf{BT}]}\right\} \\
\mathbf{p}_S^{\varepsilon_{ca}}(\mathsf{B},\mathsf{blue}^\star?) &= \mathbf{p}_S^{\varepsilon_{ca}}(\mathsf{RT},\mathsf{blue}^\star?) &= \min\left\{1, \frac{k_r}{S[\mathsf{B}]+S[\mathsf{RT}]}\right\}
\end{aligned}$$

Finally, function $\mathbf{w}_S^{\varepsilon_{ca}}$, that is used to compute the probability that a given agent receives a unicast input, associates to all the agents and all the inputs the same value 1.0. This means that all the agents have the same probability to receive a *unicast message*.

□

A NBA *specification* Σ consists of a triple

$$\langle S, \varepsilon, \Delta \rangle$$

where $\Delta \in \text{ADEF}$ is an *agents definition* while $\varepsilon \in \mathsf{Env}$ is an *environment*.

Given a system $S \in \text{SYS}$ we can consider its *scale* $k \cdot S$ where all the agents in S are replicated k times. Given a NBA *specification* $\Sigma = \langle S, \varepsilon, \Delta \rangle$, we say that ε is *scale invariant* if the capability of an agent A (in terms of probability of receiving a message and rates of actions) are not effected by the scale of a system.

Definition 1. An *environment* ε is *scale invariant* if and only if for any system S:

- for any $k \in \mathbb{N}_{>0}$, $\mathbf{r}_{k\cdot S}^{\varepsilon} = \mathbf{r}_S^{\varepsilon}$
- for any $A \in \text{AG}$ and $\alpha \in \text{ACT}$ there exits $v_{S,A,\alpha} \in \mathbb{R}_{\geq 0}$ such that:

$$\lim_{k\to\infty} \mathbf{p}_{k\cdot S}^{\varepsilon}(A,\alpha) \cdot (k \cdot S)[A] = v_{S,A,\alpha}$$

- for any $k \in \mathbb{N}_{>0}$, $\mathbf{w}_{k\cdot S}^{\varepsilon} = \mathbf{w}_S^{\varepsilon}$.

A specification $\Sigma = \langle S, \varepsilon, \Delta \rangle$ is *scale invariant* when ε is *scale invariant*.

We will see later that the use of *scale invariant environments* guarantees that the average number of agents involved in a *broadcast interaction* does not depend on/increase with the size of the population. Indeed, given a system S, $\mathbf{p}^{\varepsilon}_{S}(A,\alpha)\cdot S[A]$ is the average number of instances of A that are involved in action α. If ε is *scale invariant* this value does not depend on the scale of the system. This is because the limit of the sequence $\{\mathbf{p}^{\varepsilon}_{k\cdot S}(A,\alpha)\cdot(k\cdot S)[A]\}_{k\in\mathbb{N}_{\geq 0}}$ converges to a value $v_{S,A,\alpha}$. It is easy to see that environment ε_{ca} discussed in Example 5 is *scale invariant*.

The condition of scale invariance imposes a constraint on the model, requiring that the number of agents that can be reached by a broadcast is independent of the scale of the system. This is reasonable in most of the scenarios in which broadcast is used (e.g. voting or gossip protocols), typically because communication channels have a maximum capacity due to physical constraints on the medium in which communication happens. Hence models which violate this condition are typically describing idealised situations, in which the physical nature of communication is abstracted. An example could be a scenario of national emergency in which the government sends a text message to all the citizens. In this case the average size of the population reached will depend on the size of the country, hence violating scale invariance. These situations can be dealt with in the framework of hybrid systems; we will comment more about this in the conclusions.

2.1. Semantics

In this section we present stochastic operational semantics of NBA systems. Classically, the behaviour of (stochastic) process algebras is represented via *transition relations*. These relations, defined following a Plotkin-style, are used to infer possible computations of a process. Note that, due to *nondeterminism*, starting from the same process, different evolutions can be inferred. However, in NBA, as in other similar calculi, there is no nondeterminism as the selection of the possible next state is governed by a probability distribution.

To simplify definition of NBA semantics we will use here an approach based on FuTS style [19]. Using this approach, the behaviour of a term is described using a function that, given a *term* and a *transition label*, yields a function associating each component or system with a non-negative number. The meaning of this value depends on the context. It can be the rate of the exponential distribution characterising the time needed for the execution of the action or the probability of receiving a given broadcast/unicast message. In all cases the zero value is associated with unreachable terms.

The operational semantics of NBA systems is defined in terms of three functions that are used to compute the possible *next states* of *processes* and *systems*:

1. the function $\mathbb{P}$ that describes the behaviour of a process P;
2. the function $\mathbb{R}$ that shows how a system reacts when a message is received;
3. the function $\mathbb{S}$ that describes possible synchronisations occurring in a system.

$$\mathbb{P}(\mathbf{nil},\alpha)=\emptyset \ \ (\textsc{Nil}) \qquad \mathbb{P}(\beta.A,\alpha)=\begin{cases}\chi_A & \text{if } \alpha=\beta\\ \emptyset & \text{otherwise}\end{cases} \quad (\textsc{Act})$$

$$\frac{\mathbb{P}(P_1,\alpha)=\mathscr{A}_1 \quad \mathbb{P}(P_1,\alpha)=\mathscr{A}_2}{\mathbb{P}(P_1 \ \oplus_p \ P_2,\alpha)=\frac{p\cdot\mathscr{A}_1+(1-p)\cdot\mathscr{A}_2}{p\cdot\oplus\mathscr{A}_1+(1-p)\cdot\oplus\mathscr{A}_2}} \ (\textsc{Choice})$$

Table 1: Process semantics

Process semantics. The semantics of processes in PROC is defined via the function $\mathbb{P}:\textsc{Proc}\times\textsc{Act}\to \mathit{Dist}(\textsc{Ag})$ that, given a process term P, yields a function associating each action α with a (sub-)probability distribution over agent identifiers. Intuitively, if $\mathbb{P}(P,\alpha)(A)=p$ it means that when P executes action α, agent A is reached with probability p.

Function $\mathbb{P}$ is formally defined in Table 1. There we use $\emptyset$ to denote the (sub)-probability distribution associating 0 to each agent identifier; χ_A denotes the *dirac*-distribution associating 1 to A and 0 to all the other identifiers. Moreover, the following notation is used:

Notation 1. Let $\mathscr{X}_1,\mathscr{X}_2:X\to\mathbb{R}$, for some set X, and $v\in\mathbb{R}$, we have:

- $v\cdot\mathscr{X}_i$ denotes the function associating to each $x\in X$ the value $v\cdot\mathscr{X}_i(x)$;
- $\oplus\mathscr{X}_i$ denotes $\sum_{x\in X}\mathscr{X}_i(x)$;
- $\mathscr{X}_1+\mathscr{X}_2$ denotes the function associating to each $x\in X$ the value $\mathscr{X}_1(x)+\mathscr{X}_2(x)$.
- $\frac{\mathscr{X}_i}{v}$ is evaluated to 0 if $v=0$ and to $\frac{1}{v}\cdot\mathscr{X}_i$ otherwise.

Rules in Table 1 state that process **nil** is not able to execute any action and that when $\beta.A$ executes β it reaches the agent A with probability 1. The behaviour of $P_1 \ \oplus_p \ P_2$ needs more attention. This is obtained by the combination of behaviours of P_1 and P_2 weighted according to p and $(1-p)$ respectively. The renormalisation is needed to correctly compute the probabilities when P_1 or P_2 are not able to execute action α.

Example 6. Function $\mathbb{P}$ allows us to associate each agent A with a *probabilistic behaviour* that describes how an agent evolves when an action α is executed. The behaviour of agents of Example 4, as induced by $\mathbb{P}$, is depicted in Figure 1. In that figure *solid edges* describe action executions, while *dashed edges* describe *probability distributions*. The latter are omitted when, with probability 1.0, a single agent can be reached.

For instance, we can observe that when agent R executes action $\mathsf{red}^\star?$ it evolves with probability p_t to RT and with probability $1-p_t$ to R. Let us consider the definition of R in Example 4:

$$\mathsf{R} \ \triangleq \ \mathsf{red}^\star!.\mathsf{R} \ \oplus \ \mathsf{red}^\star?.(\mathsf{RT}\oplus_{p_t}\mathsf{R}) \ \oplus \ \mathsf{beBlue}?.\mathsf{B}$$

By using rules (ACT) and (CHOICE) we can prove that:

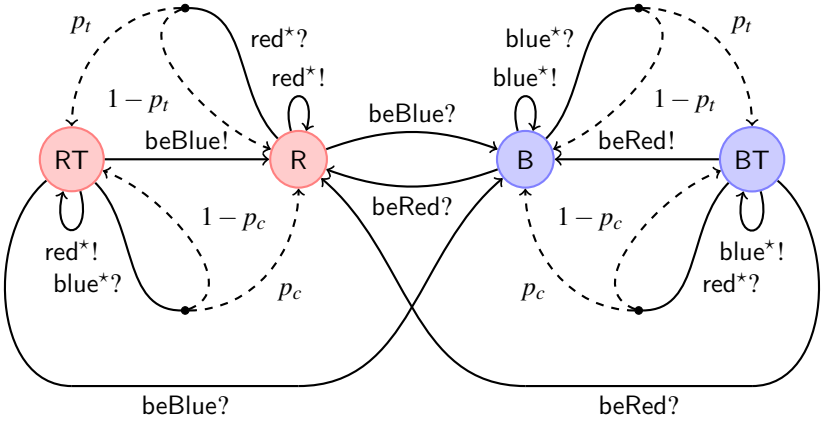

Figure 1: Probabilistic behaviour of agents in Example 4.

- $\mathbb{P}(\ \mathsf{red}^\star!.\mathsf{R}\ ,\ \mathsf{red}^\star?\ ) = \emptyset$;
- $\mathbb{P}(\ \mathsf{red}^\star?.(\mathsf{RT} \oplus_{p_t} \mathsf{R})\ ,\ \mathsf{red}^\star?\ ) = [\mathsf{RT} \mapsto p_t, \mathsf{R} \mapsto 1-p_t]$;
- $\mathbb{P}(\ \mathsf{beBlue}?.\mathsf{B}\ ,\ \mathsf{red}^\star?\ ) = \emptyset$.

where $[\mathsf{RT} \mapsto p_t, \mathsf{R} \mapsto 1-p_t]$ denotes the probability distribution associating p_t to RT and $1-p_t$ to R.

Hence, by applying rule (CHOICE) we have that:

$$\begin{aligned}\mathbb{P}(\mathsf{red}^\star!.\mathsf{R} \oplus \mathsf{red}^\star?.(\mathsf{RT} \oplus_{p_t} \mathsf{R}) \oplus \mathsf{beBlue}?.\mathsf{B}, \mathsf{red}^\star?) & \\ = \emptyset + [\mathsf{RT} \mapsto p_t, \mathsf{R} \mapsto 1-p_t] + \emptyset & \\ = [\mathsf{RT} \mapsto p_t, \mathsf{R} \mapsto 1-p_t] & \end{aligned}$$

□

Lemma 1. *For any $P \in$ PROC and $\alpha \in$ ACT the following hold:*

1. *the set $\{A \mid \mathbb{P}(P,\alpha)(A) > 0\}$ is finite;*
2. $\oplus\mathbb{P}(P,\alpha) \in \{0,1\}$.

PROOF. The proof proceeds by induction on the syntax of $P \in$ PROC.

Base of Induction. We have to consider two cases: $P = \mathbf{nil}$ and $P = \beta.B$.

($P = \mathbf{nil}$) Let us consider $\mathbb{P}(\mathbf{nil}, \alpha)$, for $\alpha \in$ ACT. By rule (NIL) of Table 1 we have that $\mathbb{P}(\mathbf{nil}, \alpha) = \emptyset$, where $\emptyset$ denotes the 0-constant function. This means that:

1. $\{A \mid \mathbb{P}(\mathbf{nil}, \alpha)(A) > 0\}$ is the empty set (that is finite);
2. $\oplus\mathbb{P}(P,\alpha) = 0$.

($P = \beta.B$) According to the semantics of Table 1, rule (ACT) must be applied to compute $\mathbb{P}(\beta.B, \alpha)$. We have to consider two cases $\alpha = \beta$ and $\alpha \neq \beta$. In the latter case $\mathbb{P}(\beta.B, \alpha) = \emptyset$ and the proof proceeds like in the case of **nil**.

When $\alpha = \beta$ we have that $\mathbb{P}(\beta.B, \alpha) = \chi_B$ and therefore:

1. $\{A \mid \mathbb{P}(P, \alpha)(A) > 0\} = \{B\}$ is finite; and
2. $\oplus\mathbb{P}(P, \alpha) = 1$.

Inductive hypothesis. Let $P_1, P_2 \in Proc$ be such that for any $\alpha \in$ ACT:

1. the set $\{A \mid \mathbb{P}(P_i, \alpha)(A) > 0\}$ is finite;
2. $\oplus\mathbb{P}(P_i, \alpha) \in \{0, 1\}$.

Inductive step. Let $P = P_1 \ \oplus_p \ P_2$. According to rule (CHOICE) of Table 1 we have that

$$\mathbb{P}(P_1 \ \oplus_p \ P_2, \alpha) = \frac{p \cdot \mathscr{A}_1 + (1-p) \cdot \mathscr{A}_1}{p \cdot \oplus\mathscr{A}_1 + (1-p) \cdot \oplus\mathscr{A}_2}$$

where $\mathbb{P}(P_1, \alpha) = \mathscr{A}_1$ and $\mathbb{P}(P_2, \alpha) = \mathscr{A}_2$. We can first observe that:

$$\{A \mid \mathbb{P}(P_1 \ \oplus_p \ P_2, \alpha)(A) > 0\} = \{A \mid \mathbb{P}(P_1, \alpha)(A) > 0\} \cup \{A \mid \mathbb{P}(P_2, \alpha)(A) > 0\}$$

which, by inductive hypothesis, is finite. Moreover:

$$\begin{aligned}
\oplus\mathbb{P}(P_1 \ \oplus_p \ P_2, \alpha) &= \sum_{A \in \mathrm{AG}} \frac{p \cdot \mathscr{A}_1(A) + (1-p) \cdot \mathscr{A}_2(A)}{p \cdot \oplus\mathscr{A}_1 + (1-p) \cdot \oplus\mathscr{A}_2} \\
&= \frac{p \cdot (\sum_{A \in \mathrm{AG}} \mathscr{A}_1(A)) + (1-p) \cdot (\sum_{A \in \mathrm{AG}} \mathscr{A}_2(A))}{p \cdot \oplus\mathscr{A}_1 + (1-p) \cdot \oplus\mathscr{A}_2} \\
&= \frac{p \cdot \oplus\mathscr{A}_1 + (1-p) \cdot \oplus\mathscr{A}_2}{p \cdot \oplus\mathscr{A}_1 + (1-p) \cdot \oplus\mathscr{A}_2}
\end{aligned}$$

For this reason, $\oplus\mathbb{P}(P, \alpha)$ is 0 when both $\oplus\mathscr{A}_1$ and $\oplus\mathscr{A}_2$ are 0 ($\frac{x}{y}$ denotes 0 whenever $y = 0$), and it is 1 when at least one of $\oplus\mathscr{A}_1$ and $\oplus\mathscr{A}_2$ is 1. By inductive hypothesis $\oplus\mathscr{A}_i \in \{0, 1\}$. Hence, $\mathbb{P}(P_1 \ \oplus_p \ P_2, \alpha) \in \{0, 1\}$.

□

Receiving message. The second step to define the NBA operational semantics consists of the definition of function $\mathbb{R}$ that shows how a system reacts when a message is received. This function is parametrised with respect to agent definition Δ and environment ε. Function $\mathbb{R}^{\Delta}_{\langle \mathbf{r}, \mathbf{p}, \mathbf{w} \rangle}$ is formally defined in Table 2 where χ_S denotes the *dirac*-distribution associating 1 to S and 0 to all the other systems and the notations of Notation 1 are used.

Rule (U-NIL) is similar to rule (NIL) of Table 1 and states that an empty system cannot receive any unicast message while rule (B-NIL) states that **0** always receives

$$\frac{}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(\mathbf{0},a?) = \emptyset}\ (\text{U-NIL}) \qquad \frac{}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(\mathbf{0},a^{\star}?) = \boldsymbol{\chi_0}}\ (\text{B-NIL})$$

$$\frac{\mathbb{P}(\Delta(A),a?) = \mathscr{A}}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(A,a?) = \mathbf{w}(A,a?)\cdot\mathscr{A}}\ (\text{U-AGENT})$$

$$\frac{\mathbb{P}(\Delta(A),a^{\star}?) = \mathscr{A}}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(A,a^{\star}?) = \mathbf{p}(A,a^{\star}?)\cdot\mathscr{A} + (1-\mathbf{p}(A,a^{\star}?)\cdot\oplus\mathscr{A})\cdot\chi_A}\ (\text{B-AGENT})$$

$$\frac{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_1,a?) = \mathscr{S}_1 \quad \mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_2,a?) = \mathscr{S}_2}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_1 \parallel S_2,a?) = \mathscr{S}_1 \parallel S_2 + S_1 \parallel \mathscr{S}_2}\ (\text{U-PAR})$$

$$\frac{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_1,a^{\star}?) = \mathscr{S}_1 \quad \mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_2,a^{\star}?) = \mathscr{S}_2}{\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(S_1 \parallel S_2,a^{\star}?) = \mathscr{S}_1 \parallel \mathscr{S}_2}\ (\text{B-PAR})$$

Table 2: System input semantics

a broadcast message and remains the same. Rules (U-AGENT) and (B-AGENT) show how a single agent can react when it receives a message via unicast or broadcast communication, respectively. In the first case, the behaviour of A is obtained from the process $\Delta(A)$, i.e. the definition of A, multiplied by the *weight* of action $a?$ for A, that is $\mathbf{w}(A,a?)$. The latter will be used to compute the probability that this specific agent will actually receive the message. When a broadcast input is considered, we have that the agent will receive the message with probability $\mathbf{p}(A,a^{\star}?)$. In that case the possible outcomes are exactly the ones of $\Delta(A)$, say $\mathscr{A}$. The same agent can ignore the input with probability $1-\mathbf{p}(A,a^{\star}?)$. Note that, to be sure that the result of $\mathbb{R}^{\Delta}_{\langle \mathbf{r},\mathbf{p},\mathbf{w}\rangle}(A,a^{\star}?)$ sums to 1, the final outcome of the rule is $\mathbf{p}(A,a^{\star}?)\cdot\mathscr{A} + (1-\mathbf{p}(A,a^{\star}?)\cdot\oplus\mathscr{A})\cdot\chi_A$. This is because when A cannot execute action $a^{\star}?$, $\oplus\mathscr{A} = 0$ and the above formula is equal to χ_A.

Rule (U-PAR) states that in $S_1 \parallel S_2$ a unicast message can be received either by S_1 or by S_2, while in rule (B-PAR) we have that a message can be received by both S_1 and S_2. In these rules the following notations are used:

- $\mathscr{S} \parallel S$ (resp. $S \parallel \mathscr{S}$) denotes the function associating $\mathscr{S}(S')$ to each system of the form $S' \parallel S$ (resp. $S \parallel S'$) and 0 to all the others;
- $\mathscr{S}_1 \parallel \mathscr{S}_2$ denotes the function that associates to each system of the form $S_1 \parallel S_2$ the value $\mathscr{S}_1(S_1)\cdot\mathscr{S}_2(S_2)$ and 0 to all the other terms.

The following lemma guarantees three *good* properties of function $\mathbb{R}$: possible configurations reachable when a message is received are finite; when α is a *broadcast input*, $\mathbb{R}_{\varepsilon(S)}(S,\alpha)$ is a probability distribution; this distribution consists of the appropriate composition of *binomial distributions*.

Lemma 2. *For each $S_1, S_2 \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \mathsf{Env}$, $\alpha \in \text{ACT}$ and $a \in \text{CH}$, the following hold:*

1. $\{S \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1, \alpha)(S) > 0\}$ *is finite;*
2. $\oplus\mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1, a^{\star}?) = 1$*;*
3. *for all $k \in \mathbb{N}$ and $A \in \text{AG}$ such that $A = P$, $\oplus\mathbb{P}(P, a^{\star}?) = 1$, and $p = \mathbf{p}^{\varepsilon}_{S_1}(A, a^{\star}?)$*

$$\mathbb{R}^{\Delta}_{\varepsilon(S_1)}(A[k], a^{\star}?) = \sum_{i=0}^{k} \binom{k}{i} \cdot \left[A[k-i] \mapsto (1-p)^{k-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^{\star}?)^i$$

where for any function $\mathscr{S} : \text{SYS} \to \mathbb{R}$, $\mathscr{S}^i$ (with $i \in \mathbb{N}$) denotes $[\mathbf{0} \mapsto 1]$ when $i = 0$ and $\mathscr{S} \parallel \mathscr{S}^{i-1}$ when $i > 0$.

4. *for all $k \in \mathbb{N}$ and $A \in \text{AG}$ such that $A = P$ and $w = \mathbf{w}^{\varepsilon}_{S_1}(A, a?)$*

$$\mathbb{R}^{\Delta}_{\varepsilon(S_1)}(A[k+1], a?) = A[k] \parallel ((k+1) \cdot w \cdot \mathbb{P}(P, a?))$$

PROOF. **Item (1):** We prove that for each $S_1, S_2 \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \mathsf{Env}$ and $\alpha \in \text{ACT}$, $\{S \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1, \alpha)(S) > 0\}$ is finite. The proof proceeds by induction on the structure of S_1.

Base of Induction. We have to consider two base cases: $S_1 = \mathbf{0}$ and $S_1 = A$.

($S_1 = \mathbf{0}$) In this case we have that either rule (U-NIL) or rule (B-NIL) is applied to compute $\mathbb{R}^{\Delta}_{\varepsilon(S_2)}(\mathbf{0}, \alpha)$. In both the cases $\{S \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(\mathbf{0}, \alpha)(S) > 0\}$ is finite since it is either the empty set or the set $\{\mathbf{0}\}$, respectively.

($S = A$) In this case the statement follows directly from Lemma 1. Indeed, either rule (U-AGENT) or rule (B-AGENT) can be applied. In the first case $\mathbb{R}^{\Delta}_{\varepsilon(S_2)}(A, a?) = \mathbf{w}^{\varepsilon}_{S_2}(A, a?) \cdot \mathscr{A}$, where $\mathbb{P}(\Delta(A), a?) = \mathscr{A}$. Moreover, by Lemma 1 we have that $\{B \mid \mathscr{A}(B) > 0\}$ is finite. The same holds for $\{B \mid \mathbf{w}^{\varepsilon}_{S_2}(B, a?) \cdot \mathscr{A}(B) > 0\}$ that is equal to $\{S \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(A, a?) > 0\}$. Similar considerations apply when rule (B-AGENT) is used.

Inductive Hypothesis. For any $S_1^1, S_1^2, S_2 \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \mathsf{Env}$ and $\alpha \in \text{ACT}$, $\{S \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1^i, \alpha)(S) > 0\}$ is finite.

Inductive Step. We have to show that $\{S_1^1 \parallel S_1^2 \mid \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1^i, \alpha)(S) > 0\}$. This follows directly from the *inductive hypothesis* and by rules in Table 2 by observing that for any $\mathscr{S}_1$ and $\mathscr{S}_2$, such that $\{S | \mathscr{S}_i(S) > 0\}$ is finite, and for any $S' \in \text{SYS}$ holds that:

- $\{S \mid (\mathscr{S}_1 \parallel \mathscr{S}_2)(S) > 0\}$ is finite;
- both $\{S \mid (\mathscr{S}_1 \parallel S')(S) > 0\}$ and $\{S \mid (S' \parallel \mathscr{S}_1)(S) > 0\}$ are finite.

Item (2): We have to prove that for each $S_1, S_2 \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \mathsf{Env}$, and $a \in \text{CH}$, $\oplus\mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1, a^{\star}?) = 1$.

The proof proceeds by induction on S_1.

Base of Induction. We have to consider two base cases: $S_1 = \mathbf{0}$ and $S_1 = A$.

($S_1 = \mathbf{0}$) By applying rule (B-NIL) of Table 2 we have that:

$$\oplus \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(\mathbf{0}, a^\star?) = \oplus \chi_{\mathbf{0}} = 1$$

($S_1 = A$) In this case we can apply rule (B-AGENT) of Table 2 to get:

$$\oplus \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(A, a^\star?) = \oplus(\mathbf{p}(A, a^\star?) \cdot \mathscr{A} + (1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \chi_A)$$

where $\mathscr{A} = \mathbb{P}(\Delta(A), a^\star?)$. We have that:

$$\begin{array}{rl} & \oplus(\mathbf{p}(A, a^\star?) \cdot \mathscr{A} + (1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \chi_A) \\ = & \oplus(\mathbf{p}(A, a^\star?) \cdot \mathscr{A}) + \oplus((1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \chi_A) \\ = & \mathbf{p}(A, a^\star?) \cdot \oplus(\mathscr{A}) + (1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \oplus \chi_A \end{array}$$

We also have that $\oplus \chi_A = 1$ and, by Lemma 1, $\oplus(\mathscr{A})$ is either 0 or 1. In the first case it follows that:

$$\mathbf{p}(A, a^\star?) \cdot \oplus(\mathscr{A}) + (1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \oplus \chi_A = \oplus \chi_A = 1$$

while when $\oplus \mathscr{A} = 1$ we have that:

$$\begin{array}{rcl} \mathbf{p}(A, a^\star?) \cdot \oplus(\mathscr{A}) + (1 - \mathbf{p}(A, a^\star?) \cdot \oplus \mathscr{A}) \cdot \oplus \chi_A & = & \\ \mathbf{p}(A, a^\star?) + (1 - \mathbf{p}(A, a^\star?)) & = & 1 \end{array}$$

Inductive Hypothesis. For any $S_1^1, S_1^2, S_2 \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \text{Env}$ and $a \in \text{CH}$, $\oplus \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1^i, a^\star?) = 1$.

Inductive Step. We have to show that $\oplus \mathbb{R}^{\Delta}_{\varepsilon(S_2)}(S_1^1 \parallel S_1^2, a^\star?) = 1$. This follows directly from the *inductive hypothesis* and by rule (B-PAR) in Table 2 by observing that for any $\mathscr{S}_1$ and $\mathscr{S}_2$, such that $\oplus \mathscr{S}_i = 1$, $\oplus(\mathscr{S}_1 \parallel \mathscr{S}_2) = 1$. The latter can be proved by observing that:

$$\begin{array}{rcl} \oplus(\mathscr{S}_1 \parallel \mathscr{S}_2) & = & \displaystyle\sum_{S \in \text{SYS}} (\mathscr{S}_1 \parallel \mathscr{S}_2)(S) \\ & = & \displaystyle\sum_{S' \in \text{SYS}} \sum_{S'' \in \text{SYS}} \mathscr{S}_1(S') \cdot \mathscr{S}_2(S'') \\ & = & \displaystyle\sum_{S' \in \text{SYS}} \mathscr{S}_1(S') \cdot \sum_{S'' \in \text{SYS}} \mathscr{S}_2(S'') \\ & = & \displaystyle\sum_{S' \in \text{SYS}} \mathscr{S}_1(S') \cdot \oplus \mathscr{S}_2 \\ & = & \displaystyle\left(\sum_{S' \in \text{SYS}} \mathscr{S}_1(S') \right) \cdot \oplus \mathscr{S}_2 \\ & = & (\oplus \mathscr{S}_1) \cdot (\oplus \mathscr{S}_2) = 1 \end{array}$$

Item (3): We can now prove that for each $S \in \text{SYS}$, agents definition $\Delta \in \text{ADEF}$, environment $\varepsilon \in \mathsf{Env}$, $a \in \text{CH}$, $k \in \mathbb{N}$ and $A \in \text{AG}$ such that $A = P$, $\oplus\mathbb{P}(P, a^\star?) = 1$, and $p = \mathbf{p}^{\varepsilon}_{S}(A, a^\star?)$:

$$\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[k], a^\star?) = \sum_{i=0}^{k} \binom{k}{i} \cdot \left[A[k-i] \mapsto p^{k-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i$$

The proof proceeds by induction on k.

Base of Induction ($k = 0$). When $k = 0$, $k \cdot A = \mathbf{0}$ and the statement follows directly from the fact that $\mathbb{R}^{\Delta}_{\varepsilon(S)}(\mathbf{0}, a^\star?) = \chi_{\mathbf{0}}$ and from the fact that $\mathbb{P}(P, a^\star?)^0 = \chi_{\mathbf{0}}$.

Inductive Hypothesis ($k \leq n$). Let us assume that for any $k \leq n$

$$\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[k], a^\star?) = \sum_{i=0}^{k} \binom{k}{i} \cdot \left[A[k-i] \mapsto p^{k-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i$$

Inductive Step ($k = n+1$). Let us consider $A[n+1]$. By rule (B-PAR) we have that:

$$\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[n+1], a^\star?) = \mathbb{R}^{\Delta}_{\varepsilon(S)}(A \parallel A[n], a^\star?) = \mathbb{R}^{\Delta}_{\varepsilon(S)}(A, a^\star?) \parallel \mathbb{R}^{\Delta}_{\varepsilon(S)}(A[n], a^\star?)$$

By rule (B-AGENT), and from the fact that $\oplus\mathbb{P}(P, a^\star?) = 1$, we have that:

$$\mathbb{R}^{\Delta}_{\varepsilon(S)}(A, a^\star?) = p \cdot \mathbb{P}(P, a^\star?) + (1-p) \cdot \chi_A$$

By inductive hypothesis we have that:

$$\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[n], a^\star?) = \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i$$

Hence:

$$
\begin{aligned}
&\mathbb{R}^{\Delta}_{\varepsilon(S)}(A, a^\star?) \parallel \mathbb{R}^{\Delta}_{\varepsilon(S)}(A[n], a^\star?) \\
&\quad = (p \cdot \mathbb{P}(P, a^\star?) + (1-p) \cdot \chi_A) \parallel \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\quad = (p \cdot \mathbb{P}(P, a^\star?)) \parallel \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\qquad + ((1-p) \cdot \chi_A) \parallel \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\quad = \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^{i+1} \cdot \mathbb{P}(P, a^\star?)^{i+1} \\
&\qquad + \sum_{i=0}^{n} \binom{n}{i} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\quad = \binom{n}{n} \left[A[0] \mapsto (1-p)^0\right] \parallel p^{n+1} \cdot \mathbb{P}(P, a^\star?)^{n+1} \\
&\qquad + \sum_{i=0}^{n-1} \binom{n}{i} \cdot \left[A[n-i] \mapsto (1-p)^{n-i}\right] \parallel p^{i+1} \cdot \mathbb{P}(P, a^\star?)^{i+1} \\
&\qquad + \sum_{i=1}^{n} \binom{n}{i} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\qquad + \binom{n}{0} \cdot \left[A[n+1] \mapsto (1-p)^{n+1}\right] \parallel p^0 \cdot \mathbb{P}(P, a^\star?)^0 \\
&\quad = \binom{n+1}{n+1} \left[A[0] \mapsto (1-p)^0\right] \parallel p^{n+1} \cdot \mathbb{P}(P, a^\star?)^{n+1} \\
&\qquad + \sum_{i=1}^{n} \binom{n}{i-1} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\qquad + \sum_{i=1}^{n} \binom{n}{i} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\qquad + \binom{n+1}{0} \cdot \left[A[n+1] \mapsto (1-p)^{n+1}\right] \parallel p^0 \cdot \mathbb{P}(P, a^\star?)^0 \\
&\quad = \binom{n+1}{n+1} \left[A[0] \mapsto (1-p)^0\right] \parallel p^{n+1} \cdot \mathbb{P}(P, a^\star?)^{n+1} \\
&\qquad + \sum_{i=1}^{n} \left(\binom{n}{i-1} + \binom{n}{i} \right) \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\qquad + \binom{n+1}{0} \cdot \left[A[n+1] \mapsto (1-p)^{n+1}\right] \parallel p^0 \cdot \mathbb{P}(P, a^\star?)^0
\end{aligned}
$$

$$
\begin{aligned}
&= \binom{n+1}{n+1} \left[A[0] \mapsto (1-p)^0\right] \parallel p^{n+1} \cdot \mathbb{P}(P, a^\star?)^{n+1} \\
&\quad + \sum_{i=1}^{n} \binom{n+1}{i} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i \\
&\quad + \binom{n+1}{0} \cdot \left[A[n+1] \mapsto (1-p)^{n+1}\right] \parallel p^0 \cdot \mathbb{P}(P, a^\star?)^0 \\
&= \sum_{i=0}^{n+1} \binom{n+1}{i} \cdot \left[A[n+1-i] \mapsto (1-p)^{n+1-i}\right] \parallel p^i \cdot \mathbb{P}(P, a^\star?)^i
\end{aligned}
$$

Item (4): We have to prove that, for each $S_1, S_2 \in \textsc{Sys}$, agents definition $\Delta \in \textsc{ADef}$, environment $\varepsilon \in \mathsf{Env}$, $a \in \textsc{Ch}$, and for all $k \in \mathbb{N}$ and $A \in \textsc{Ag}$ such that $A = P$ and $w = \mathbf{w}_S^{\varepsilon}(A, a?)$

$$
\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[k+1], a?) = A[k] \parallel ((k+1) \cdot w \cdot \mathbb{P}(P, a?))
$$

As for the previous case, the proof proceeds by induction on k.

Base of Induction $(k = 0)$. Directly from rule (U-AGENT) of Table 2, we have that

$$
\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[1], a?) = w \cdot \mathbb{P}(P, a?)) = \mathbf{0} \parallel w \cdot \mathbb{P}(P, a?)) = A[0] \parallel w \cdot \mathbb{P}(P, a?)
$$

Inductive Hypothesis $(k \leq n)$. Let us assume that for any $k \leq n$

$$
\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[k+1], a?) = A[k] \parallel (k+1) \cdot w \cdot \mathbb{P}(P, a?)
$$

Inductive Step $(k = n+1)$. We have that:

$$
\begin{aligned}
\mathbb{R}^{\Delta}_{\varepsilon(S)}(A[(n+1)+1], a?) &= \mathbb{R}^{\Delta}_{\varepsilon(S)}(A \parallel A[(n+1)], a?) \\
&\qquad \text{By applying (U-PAR) of Table 2} \\
&= \mathbb{R}^{\Delta}_{\varepsilon(S)}(A, a?) \parallel A[(n+1)] + A \parallel \mathbb{R}^{\Delta}_{\varepsilon(S)}(A[(n+1)], a?) \\
&\qquad \text{By applying (U-AGENT) of Table 2} \\
&\qquad \text{and by Inductive Hypothesis} \\
&= (w \cdot \mathbb{P}(P, a?))) \parallel A[(n+1)] \\
&\quad + A \parallel A[n] \parallel ((n+1) \cdot w \cdot \mathbb{P}(P, a?)) \\
&= (w \cdot \mathbb{P}(P, a?))) \parallel A[(n+1)] \\
&\quad + A[n+1] \parallel ((n+1) \cdot w \cdot \mathbb{P}(P, a?)) \\
&= A[n+1] \parallel (w \cdot \mathbb{P}(P, a?) + (n+1) \cdot w \cdot \mathbb{P}(P, a?)) \\
&= A[n+1] \parallel (n+2) \cdot w \cdot \mathbb{P}(P, a?)
\end{aligned}
$$

□

Given a system S, and an *environment* ε, the number of instances of an agent A involved in an input $a^\star?$ depends on the probability $\mathbf{p}^{\varepsilon}_{S}(A, a^\star?)$. Indeed, when $S(A) = n > 0$ we have that on the average $\mathbf{p}(A, a^\star?) \cdot n$ instances of A will receive a broadcast input on channel a. Note that, if ε is *scale invariant* (see Definition 1), the average number of agents that receive a broadcast message does not change if we scale/increase the size of the system.

Example 7. Let Δ_{ca} be the agents definition of Example 4, ε_{ca} be the environment of Example 5, which associates each input action with the same weight 1.0, and S be the system defined below:

$$S = \mathsf{R}[k_1] \parallel \mathsf{B}[k_2] \parallel \mathsf{RT}[k_3] \parallel \mathsf{BT}[k_4]$$

We can use function $\mathbb{R}^{\Delta_{ca}}_{\varepsilon_{ca}(S)}$ to compute how this system can react when a beRed? is performed. We can observe (see Figure 1) that:

- $\mathbb{P}(\Delta_{ca}(\mathsf{R}), \mathsf{beRed?}) = \mathbb{P}(\mathsf{RT}, \mathsf{beRed?}) = \emptyset$
- $\mathbb{P}(\Delta_{ca}(\mathsf{B}), \mathsf{beRed?}) = [\mathsf{R} \mapsto 1.0]$
- $\mathbb{P}(\Delta_{ca}(\mathsf{BT}), \mathsf{beRed?}) = [\mathsf{R} \mapsto 1.0]$

Moreover, whenever $k_2 > 0$ and $k_4 > 0$, directly from rule (U-AGENT) in Table 2 and Lemma 2 (4), we have that:

- $\mathbb{R}^{\Delta_{ca}}_{\varepsilon_{ca}(S)}(\mathsf{R}[k_1], \mathsf{beRed?}) = \emptyset$
- $\mathbb{R}^{\Delta_{ca}}_{\varepsilon_{ca}(S)}(\mathsf{B}[k_2], \mathsf{beRed?}) = \mathsf{B}[k_2 - 1] \parallel k_2 \cdot [\mathsf{R} \mapsto 1.0] = \mathsf{B}[k_2 - 1] \parallel [\mathsf{R} \mapsto k_2] =$

  $$[\mathsf{B}[k_2 - 1] \parallel \mathsf{R} \mapsto k_2]$$

- $\mathbb{R}^{\Delta_{ca}}_{\varepsilon_{ca}(S)}(\mathsf{RT}[k_3], \mathsf{beRed?}) = \emptyset$
- $\mathbb{R}^{\Delta_{ca}}_{\varepsilon_{ca}(S)}(\mathsf{BT}[k_4], \mathsf{beRed?}) = \mathsf{BT}[k_4 - 1] \parallel k_4 \cdot [\mathsf{R} \mapsto 1] == \mathsf{BT}[k_4 - 1] \parallel [\mathsf{R} \mapsto k_4]$

  $$[\mathsf{R} \parallel \mathsf{BT}[k_4 - 1] \mapsto k_4]$$

Finally, by multiple applications of rule (U-PAR) we have that:

$$\begin{array}{rcl} \mathbb{R}_{\varepsilon_{ca}(S)}(S, \mathsf{beRed?}) & = & [\\ & & \quad (\mathsf{R}[k_1 + 1] \parallel \mathsf{B}[k_2 - 1] \parallel \mathsf{RT}[k_3] \parallel \mathsf{BT}[k_4]) \mapsto k_2, \\ & & \quad (\mathsf{R}[k_1 + 1] \parallel \mathsf{B}[k_2] \parallel \mathsf{RT}[k_3] \parallel \mathsf{BT}[k_4 - 1]) \mapsto k_4 \\ & &] \end{array}$$

This describes the fact that action beRed? can be executed either by an agent B or by an agent BT. In both cases the receiving agent becomes an R. Moreover, values k_2 and

k_4 identify the *weights* of these possible transitions and will be used to compute the probability that the action is performed by an instance of B or by an agent BT.

Recall a *unicast* message can be received only by one agent. This is selected via a probability that depends on the *weights* associated to *possible receivers*. In contrast, *broadcast* messages can be received by multiple agents at the same time. The exact number of receivers is controlled by the environment Δ that, via $\mathbf{p}_\Delta$, gives the probability that an agent A receives a broadcast message $\alpha^\star?$. As already observed in Lemma 2, we have that the probability that n instances of agent A are involved in a broadcast interaction is governed by a *binomial distribution*.

In Example 5 we have assumed that k agents can receive a broadcast message on average. If we consider action $\mathsf{red}^\star?$ in configuration S above, each agent R will receive the message with probability $p_\mathsf{R} = \frac{k_r}{k_1+k_4}$, while each instance of BT will receive the same message with probability $p_\mathsf{BT} = \frac{k_r}{k_1+k_4}$. Note that R and BT are the only agents that can execute action $\mathsf{red}^\star?$ (see Figure 1):

- $\mathbb{P}(\Delta_{ca}(\mathsf{R}, \mathsf{red}^\star?)) = [\mathsf{R} \mapsto (1-p_t), \mathsf{RT} \mapsto p_t] = \mathscr{P}_\mathsf{R}$
- $\mathbb{P}(\Delta_{ca}(\mathsf{BT}, \mathsf{red}^\star?)) = [\mathsf{BT} \mapsto (1-p_c), \mathsf{B} \mapsto p_c] = \mathscr{P}_\mathsf{BT}$
- $\mathbb{P}(\Delta_{ca}(\mathsf{B}, \mathsf{red}^\star?)) = \mathbb{P}(\Delta_{ca}(\mathsf{RT}, \mathsf{red}^\star?) = \emptyset$

From Lemma 2 (3) we have that:

- $\mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{R}[k_1], \mathsf{red}^\star?)) = \sum_{n=0}^{k_1} \binom{k_1}{n} \left([\mathsf{R}[k_1-n] \mapsto (1-p_\mathsf{R})^{k_1-n}] \parallel p_\mathsf{R}^n \cdot \mathscr{P}_\mathsf{R}^n\right)$
- $\mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{B}[k_2], \mathsf{red}^\star?)) = [\mathsf{B}[k_2] \mapsto 1]$
- $\mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{RT}[k_3], \mathsf{red}^\star?) = [\mathsf{RT}[k_3] \mapsto 1]$
- $\mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{BT}[k_4], \mathsf{red}^\star?)) = \sum_{n=0}^{k_4} \binom{k_4}{n} \left([\mathsf{BT}[k_4-n] \mapsto (1-p_\mathsf{BT})^{k_1-n}] \parallel p_\mathsf{BT}^n \cdot \mathscr{P}_\mathsf{BT}^n\right)$

From the above by multiple applications of rule (B-PAR) we have that:

$$\begin{array}{rcl}
\mathbb{R}_{\varepsilon_{ca}(S)}(S, \mathsf{red}^\star?) & = & \mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{R}[k_1], \mathsf{red}^\star?)) \parallel \mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{B}[k_2], \mathsf{red}^\star?)) \\
& & \parallel \mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{RT}[k_3], \mathsf{red}^\star?) \parallel \mathbb{R}^\Delta_{\varepsilon_{ca}(S)}(\mathsf{BT}[k_4], \mathsf{red}^\star?)) \\
& = & \left(\sum_{n=0}^{k_1} \binom{k_1}{n} \left([\mathsf{R}[k_1-n] \mapsto (1-p_\mathsf{R})^{k_1-n}] \parallel p_\mathsf{R}^n \cdot \mathscr{P}_\mathsf{R}^n\right)\right) \\
& & \parallel [\mathsf{B}[k_2] \mapsto 1] \parallel [\mathsf{RT}[k_3] \mapsto 1] \\
& & \parallel \left(\sum_{n=0}^{k_4} \binom{k_4}{n} \left([\mathsf{BT}[k_4-n] \mapsto (1-p_\mathsf{BT})^{k_1-n}] \parallel p_\mathsf{BT}^n \cdot \mathscr{P}_\mathsf{BT}^n\right)\right)
\end{array}$$

Note that rules of Table 2 allow us to compute $\mathbb{R}_{\varepsilon_{ca}(S)}(S, \mathsf{beRed}?)$ following a structured approach in the spirit of *structural operational semantics*. □

Synchronisation rule. The last step to define the operational semantics of NBA is to introduce the function $\mathbb{S}$ that computes all the interactions occurring over a given action. Given a system S, and an action a, function $\mathbb{S}^{\Delta}_{\varepsilon}(S,a^{\star})$ (resp. $\mathbb{S}^{\Delta}_{\varepsilon}(S,a)$) will return a function $\mathscr{S}$ associating each system S' with a non negative real value. The latter indicates the transition rate from S to S' when a broadcast (resp. unicast) synchronisation on action $a^{\star}$ (a) is performed. Similarly to $\mathbb{R}$, function $\mathbb{S}$ is parametrised with respect to agent definitions Δ and the environment ε.

Function $\mathbb{S}^{\Delta}_{\varepsilon}$ is formally defined by the following two equations:

$$\mathbb{S}^{\Delta}_{\varepsilon}(S,a) \;=\; \sum_{A\in S}\left(S[A]\cdot \mathbf{r}(A,a)\cdot \mathbb{P}(\Delta(A),a!) \parallel \frac{\mathbb{R}^{\Delta}_{\varepsilon(S)}(S-A,a?)}{\oplus\left(\mathbb{R}^{\Delta}_{\varepsilon(S)}(S-A,a?)\right)}\right) \qquad \text{(U-SYN)}$$

$$\mathbb{S}^{\Delta}_{\varepsilon}(S,a^{\star}) \;=\; \sum_{A\in S}\left(S[A]\cdot \mathbf{r}(A,a)\cdot \mathbb{P}(\Delta(A),a^{\star}!) \parallel \mathbb{R}^{\Delta}_{\varepsilon(S)}(S-A,a^{\star}?)\right) \qquad \text{(B-SYN)}$$

The following Lemma guarantees that, in a single step, a system S can reach a finite number of configurations.

Lemma 3. *For any specification $\langle S,\varepsilon,\Delta\rangle$ and for any $a\in$ CH, the following holds:*

- $\{S' \mid \mathbb{S}^{\Delta}_{\varepsilon}(S,a)(S')>0\}$ *is finite;*
- $\{S' \mid \mathbb{S}^{\Delta}_{\varepsilon}(S,a^{\star})(S')>0\}$ *is finite.*

PROOF. The proof follows directly from Lemma 1, Lemma 2 and from the definition of rules (U-SYN) and (B-SYN).

Example 8. Let us consider again the system S of Example 7. Function $\mathbb{S}$ can be used to compute possible outcomes triggered by a *unicast* or *broadcast* synchronisation.

For instance, the result of a synchronisation over beRed! can be obtained by using rule (U-SYN) and composing

$$k_4\cdot\lambda_c\cdot\mathbb{P}(\Delta_{ca}(\mathsf{BT}),\mathsf{beRed}!) = [\mathsf{B}\mapsto k_4\cdot\lambda_c]$$

that identifies a computation associated with an output, with $\mathbb{R}^{\Delta}_{\varepsilon(S)}(S-\mathsf{BT},\mathsf{beRed}?)$, that describes how the remaining part of the system (Recall $S-\mathsf{BT}$ is the system obtained from S by removing one instance of BT) *receives* the message (see Example 7). The latter is also *renormalised* according to the *total weight* of *inputs* to compute the probability that resolves the *race* among the possible receivers, that is $w=k_2+(k_4-1)$. The result of this composition is reported below:

$$\begin{aligned}[\quad &(\mathsf{R}[k_1+1]\parallel\mathsf{B}[k_2]\parallel\mathsf{RT}[k_3]\parallel\mathsf{BT}[k_4-1])\mapsto\frac{k_2}{k_2+(k_4-1)}\cdot k_4\cdot\lambda_c,\\ &(\mathsf{R}[k_1+1]\parallel\mathsf{B}[k_2+1]\parallel\mathsf{RT}[k_3]\parallel\mathsf{BT}[k_4-2])\mapsto\frac{k_4}{k_2+(k_4-1)}\cdot k_4\cdot\lambda_c\ ]\end{aligned}$$

□

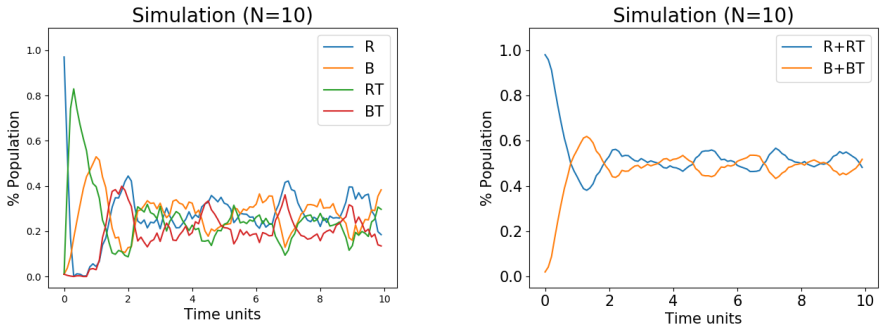

Figure 2: Simulation results (single run) of the system of Example 3 showing the fractions of agents in states R, B, RT and BT (left side) and the fractions of *red* (R or RT) and *blue* (B or BT) agents in the system (right side). Simulation parameters: $N = 10$, $\lambda_a = \lambda_c = 1.0$, $p_t = p_c = 0.5$, $k = 10$)

Given a specification $\Sigma = \langle S, \varepsilon, \Delta \rangle$, we can define the associated CTMC; this is obtained as $(\textsc{Sys}, \mathbf{Q}^{\varepsilon}_{\Delta})$ where the infinitesimal generator matrix $\mathbf{Q}^{\varepsilon}_{\Delta}$ is defined as follows:

$$\mathbf{Q}^{\varepsilon}_{\Delta}[S_1, S_2] = \sum_{a} \left(\mathbb{S}^{\Delta}_{\varepsilon}(S_1, a, S_2) + \mathbb{S}^{\Delta}_{\varepsilon}(S_1, a^{\star}, S_2) \right)$$

Example 9. Functions $\mathbb{P}$, $\mathbb{R}$ and $\mathbb{S}$ can be used as the base for implementing a *stochastic simulator* for NBA. In Figure 2 we report the results of the simulation of the system considered in Example 3 with $N = 10$. Simulations have been performed with the framework Sibilla [2], simulation code is available at the link http://bit.ly/2R2qt1E. On the left the fractions of agents in state R, B, RT and BT are reported. On the right, we focus on the fractions of *red* (R or RT) and *blue* (B or BT) agents in the system. We can observe that, even if we start from an unbalanced configuration (90% of agents are *red*), a more *balanced* state is quickly reached.

Notations and Definitions. Let $\Sigma = \langle S, \varepsilon, \Delta \rangle$ be a specification, then the following notations will be used:

- for any $A, A' \in \textsc{Ag}$, we will write $A \overset{\alpha,p}{\mapsto}_{\Delta} A'$ if and only if $\Delta(A) = P$ and $\mathbb{P}(P, \alpha)(A') = p$. We will write $A \overset{\alpha}{\mapsto}_{\Delta} A'$ if and only if there exists $p > 0$ such that $A \overset{\alpha,p}{\mapsto}_{\Delta} A'$. Moreover, we will write $A \overset{\alpha}{\mapsto}_{\Delta}$ to denote that there exists A' such that $A \overset{\alpha}{\mapsto}_{\Delta} A'$.
- for any $A \in \textsc{Ag}$, $\mathsf{actions}(\Delta, A) = \{\alpha \mid A \overset{\alpha}{\mapsto}_{\Delta}\}$;
- for any $A \in \textsc{Ag}$, $\mathsf{agents}(\Delta, A)$ is the smallest $X_{\textsc{Ag}} \subset \textsc{Ag}$ such that:
  - $A \in X_{\textsc{Ag}}$;

[2] https://github.com/quasylab/sibilla

– for any $A_1 \in X_{\mathrm{AG}}$, if $A_1 \overset{\alpha}{\mapsto}_\Delta A_2$, for some α, then $A_2 \in X_{\mathrm{AG}}$.

- for any S,
$$\mathsf{agents}(\Delta, S) = \bigcup_{A \in S} \mathsf{agents}(\Delta, A)$$
- for any S,
$$\mathsf{actions}(\Delta, S) = \bigcup_{A \in \mathsf{agents}(\Delta,S)} \mathsf{actions}(\Delta, A)$$
- We say that $\langle S, \varepsilon, \Delta \rangle$ is *well formed* if and only if $\mathsf{agents}(\Delta, S)$ is finite.

3. Population models

As the intention for NBA is to model large numbers of interacting components, we seek to give models a semantics in terms of population continuous time Markov chains (PCTMC) rather than capturing every agent individually. In a PCTMC the states keep track of how the population is distributed over the possible states.

Definition 2. A Population Continuous Time Markov Chain (PCTMC) model is a tuple $M = (\mathbf{X}, \mathscr{D}, \mathscr{T}, \mathbf{d}_0)$ where:

- $\mathbf{X} = (X_1, \ldots, X_n)$ is a vector of variables;
- each X_i takes values in a *finite* or *countable* domain $\mathscr{D}_i \subset \mathbb{R}$;
- $\mathscr{D} = \Pi_i \mathscr{D}_i$;
- $\mathbf{d}_0 \in \mathscr{D}$ is the *initial state* of the model;
- $\mathscr{T} = \{\tau_1, \ldots, \tau_m\}$ is the set of *transitions* $\tau_i = (\ell, \pi, r)$ where:
  - ℓ is the *label* of the transition;
  - $\pi : \mathscr{D} \to \mathit{Dist}_F(\mathscr{D})$ is a function associating each population state with a probability (finite) distribution with finite support over vectors $\mathbf{v}$ denoting the *update* induced by the rule;
  - $r : \mathscr{D} \to \mathbb{R}_{\geq 0}$ is a *rate function*. We require that if $r(\mathbf{d}) \neq 0$ then for all $\mathbf{v}$ such that $\pi(\mathbf{d})(\mathbf{v}) > 0$, $\mathbf{d} + \mathbf{v} \in \mathscr{D}$.

For a transition $\tau = \langle \ell, \pi, r \rangle$ we will use ℓ_τ to denote the label of τ. Similar notation will be used also for the other components of τ. Note that the outcome of an action in a state is not deterministic but governed by the probability distribution $\pi(\mathbf{d})$, thus we can also define $\mu_\tau(\mathbf{d})$ and $\mathbf{w}_\tau(\mathbf{d})$, respectively the average and the variance of the change in population due to a firing of a τ transition. Formally, these are defined as follows (with $\mathbf{v}[i]$ we denote the i-th component of vector $\mathbf{v}$):

$$\mu_\tau(\mathbf{d}) = \sum_{\mathbf{v} \in \mathscr{D}} \mathbf{v} \cdot \pi_\tau(\mathbf{d})(\mathbf{v}) \tag{3}$$

$$\mathbf{w}_\tau(\mathbf{d})[i] = \sum_{\mathbf{v} \in \mathscr{D}} \mathbf{v}[i]^2 \pi_\tau(\mathbf{d})(\mathbf{v}) - \mu_\tau(\mathbf{d})[i]^2 \tag{4}$$

In the following derivations it may be convenient to make models depend on an index N, which represents the population size. This value will be also used to index a sequence of population models M^N, where each component of M^N may depend on N (denoted by $\mathscr{D}^N$, $\mathscr{T}^N$,...).

Given a PCTMC M^N we can easily define the associated CTMC: this is obtained as $(\mathscr{D}^N, \mathbf{Q}_{\mathbf{M^N}})$ where the infinitesimal generator matrix $\mathbf{Q}_{\mathbf{M^N}}$ is defined as follows:

$$\mathbf{Q}_{\mathbf{M^N}}[\mathbf{d},\mathbf{d}'] = \sum_{\tau \in M^N} r_\tau^N(\mathbf{d}) \cdot \sum_{\mathbf{v}|\mathbf{d}+\mathbf{v}=\mathbf{d}'} \pi_\tau(\mathbf{d})(\mathbf{v})$$

In order to compare models of growing sizes of populations, it is convenient to switch to a normalised representation, where instead of population *counts* in each feasible local state, we record the *proportion* of the population that occupy each state (sometimes termed *occupancy measures*).

Let $\delta_N = \frac{1}{N}$; then given a model $M^N = (\mathbf{X}, \mathscr{D}, \mathscr{T}, \mathbf{d}_0)$, we can define a *normalising operator* $\hat{\cdot}$ as follows:

- for all $\mathbf{d} \in \mathscr{D}^N$, $\hat{\mathbf{d}} = \delta_N \cdot \mathbf{d}$;
- $\hat{\mathscr{D}}^N = \{\hat{\mathbf{d}} \mid \mathbf{d} \in \mathscr{D}^N\}$;
- for all $\tau \in (\ell, \pi^N, r^N)$, $\hat{\tau} = (\ell, \hat{\pi}^N, \hat{r}^n)$, where:
  - for any $\hat{\mathbf{d}} \in \hat{\mathscr{D}}$, $\hat{\pi}^N(\hat{\mathbf{d}})$ is a probability distribution in $\mathbb{R}^n$ defined by:

$$\hat{\pi}^N(\hat{\mathbf{d}}) = \pi^N(\frac{1}{\delta_N} \cdot \hat{\mathbf{d}})$$

  - for any $\hat{\mathbf{d}} \in \hat{\mathscr{D}}$, $\hat{r}^N(\hat{\mathbf{d}}) = r^N(\frac{1}{\delta_N} \cdot \hat{\mathbf{d}})$;
- $\hat{\mathscr{T}}^N = \{\hat{\tau^N} | \tau^N \in \mathscr{T}^N\}$.

We also define the mean increment $\hat{\mu}_\tau(\hat{\mathbf{d}})$ and the variance of the increment $\hat{\mathbf{w}}_\tau(\hat{\mathbf{d}})$ for the normalised model, as follows:

$$\begin{aligned} \hat{\mu}_\tau(\hat{\mathbf{d}}) &= \sum_{\mathbf{v} \in \mathscr{D}} (\delta_N \mathbf{v}) \cdot \hat{\pi}_\tau(\hat{\mathbf{d}})(\mathbf{v}) \\ \hat{\mathbf{w}}_\tau(\hat{\mathbf{d}})[i] &= \sum_{\mathbf{v} \in \mathscr{D}} (\delta_N \mathbf{v}[i])^2 \cdot \hat{\pi}_\tau(\hat{\mathbf{d}})(\mathbf{v}) - \hat{\mu}_\tau(\hat{\mathbf{d}})[i]^2. \end{aligned}$$

Simple algebra allows us to derive the following:

Lemma 4. *Let $M^N = (\mathbf{X}, \mathscr{D}, \mathscr{T}, \mathbf{d}_0)$ be a PCTMC and $\hat{M}^N = (\mathbf{X}, \hat{\mathscr{D}}, \hat{\mathscr{T}}, \hat{\mathbf{d}}_0)$ its normalised version, then for any $\tau \in \mathscr{T}$:*

$$\begin{aligned} \hat{\mu}_\tau(\hat{\mathbf{d}}) &= \delta_N \cdot \mu_\tau\left(\frac{1}{\delta_N} \cdot \hat{\mathbf{d}}\right) \\ \hat{\mathbf{w}}_\tau(\hat{\mathbf{d}}) &= \delta_N^2 \cdot \mathbf{w}_\tau\left(\frac{1}{\delta_N} \cdot \hat{\mathbf{d}}\right). \end{aligned}$$

Definition 3. A sequence of *normalised* PCTMC model $(\hat{M}^N)_{N>N_0}$ *admits a fluid approximation* if and only if:

- the system size grows linearly with N;
- there exists a subset $E \subseteq \mathbb{R}^n$ such that $\hat{\mathscr{D}}^N \subseteq E$ for all N;
- for each $\tau \in \mathscr{T}^N$ there exists a locally Lipschitz continuous and bounded function[3] $\mathbf{m}_\tau : E \to \mathbb{R}$ such that uniformly for $\hat{\mathbf{d}} \in E$, $\lim_{N\to\infty} \mu_\tau^N(\frac{1}{\delta_N}\hat{\mathbf{d}}) = \mathbf{m}_\tau(\hat{\mathbf{d}})$
- for each $\tau \in \mathscr{T}^N$, $\sup_N \mathbf{w}_\tau^N\left(\frac{1}{\delta_N}\cdot\hat{\mathbf{d}}\right)$ is locally bounded in E;
- there is a locally Lipschitz continuous and bounded function $g : E \to \mathbb{R}_{\geq 0}$ such that:
$$\lim_{N\to\infty} \delta_N \hat{r}^N(\hat{\mathbf{d}}) = g(\hat{\mathbf{d}})$$
uniformly for $\hat{\mathbf{d}} \in E$.

In the previous definition, it is worth pointing out that the third condition tells us that the average number of updated agents due to a transition is essentially independent of N, hence after normalisation, $\hat{\mu}_\tau(\hat{\mathbf{d}})$ converges to zero.

We let

$$F^N(\hat{\mathbf{d}}) = \sum_{\tau\in\mathscr{T}^N} \hat{\mu}_\tau^N(\hat{\mathbf{d}})\hat{r}^N(\hat{\mathbf{d}})$$

$$F(\hat{\mathbf{d}}) = \lim_{N\to\infty} F^N(\hat{\mathbf{d}}) = \sum_{\tau\in\mathscr{T}^N} \mathbf{m}_\tau(\hat{\mathbf{d}})\cdot g_\tau(\hat{\mathbf{d}})$$

The drift F defines the vector field of the deterministic limit whose trajectories $\hat{\mathbf{x}}(t)$ are solutions of the initial value problem

$$\frac{d\hat{\mathbf{x}}(t)}{dt} = F(\hat{\mathbf{x}}(t)) \qquad \text{with} \qquad \hat{\mathbf{x}}(0) = \hat{\mathbf{d}}_0 = \lim_{N\to\infty} \hat{\mathbf{d}}_0^N \in E$$

Theorem 5. *Deterministic Approximation for PCTMCs via Fluid Approximation. Let $\{\mathbf{X}^N\}_{N\geq N_0}$ be the sequence of Markov processes associated with the sequence of PCTMC models $\{M^N\}_{N\geq N_0}$ admitting a fluid approximation, $\hat{\mathbf{x}}(t)$ as above, and assume* all conditions above *are satisfied. Then, for any finite time horizon $T < \infty$, it holds that:*

$$\mathbb{P}\{\lim_{N\to\infty} \sup_{0\leq t\leq T} \| \mathbf{X}^N(t) - \hat{\mathbf{x}}(t) \| = 0\} = 1,$$

i.e. $\sup_{0\leq t\leq T} \| \mathbf{X}^N(t) - \hat{\mathbf{x}}(t) \|$ *converges to zero almost surely.*

[3] A locally Lipschitz continuous function f is Lipschitz in any compact set K: $\forall K \subset \mathbb{R}^n$ compact, there exists a constant $L_K > 0$ such that for all $\mathbf{x},\mathbf{y} \in K$, $\|f(\mathbf{x}) - f(\mathbf{y})\| \leq L_K\|\mathbf{x}-\mathbf{y}\|$. A locally bounded function f is bounded in any compact set K: $\forall K \subset \mathbb{R}^n$ compact, there exists a constant $M_K > 0$ such that $\sup_{\mathbf{x}\in K}\|f(\mathbf{x})\| \leq M_K$.

PROOF. The theorem follows by recasting the population models presented above into the framework of Darling's proof of the fluid approximation theorem of [17]. In this paper, conditions are stated in terms of the global rate (i.e. the sum of the rates of each transition) and the global second non-centred moment of the update, though they can be equivalently rewritten in terms of variance. Furthermore, as we have a finite number of transitions $\tau \in \mathscr{T}$, we can relativize these conditions for each transition.

Consider any compact set $K \subseteq E$, and let $K^N = \hat{\mathscr{D}}^N \cap K$. In this formulation, what is required is that

1. $\lim_{N\to\infty} F^N(\hat{\mathbf{d}}) = F(\hat{\mathbf{d}})$, uniformly in $\hat{\mathbf{d}}$;
2. for each compact set $K \subseteq E$, there exists a constant M_K^1 such that for each $\tau \in \mathscr{T}$:
$$\sup_{\hat{\mathbf{d}}\in K^N} \|\hat{r}_\tau^N(\hat{\mathbf{d}})\| \leq M_K^1 \cdot N;$$
3. for each compact set $K \subseteq E$, there exists a constant M_K^2 such that for each $\tau \in \mathscr{T}$:
$$\sup_{\hat{\mathbf{d}}\in K^N} \|\hat{\mathbf{w}}_\tau^N(\hat{\mathbf{d}})\| \leq \delta_N^2 \cdot M_K^2;$$

Now, Condition 1 above follows from the convergence of the mean update and of the rate functions. Condition 2 follows from the convergence of the rate functions, computing M_K^1 as the upper bound in K of the continuous function g_τ. Condition 3 follows from Lemma 4, and using the boundedness of non-normalised second non-centred moment, as for Definition 3.

4. NBA Population Semantics

In this section we present a semantics of NBA specifications as population models. Give a specification $\Sigma = \langle S, \varepsilon, \Delta\rangle$ we show how we can build a population $M_\Sigma = (\mathbf{X}_\Sigma, \mathscr{D}_\Sigma, \mathscr{T}_\Sigma, \mathbf{d}_\Sigma)$ that has exactly the same *stochastic behaviour* as Σ.

Let $\mathsf{agents}(\Delta, S) = \{A_1, \ldots, A_k\}$. The *vector of variables* $\mathbf{X}_\Sigma$ of M_Σ consists of the set of *variables* X_{A_i} taking values in the domain $\mathbb{N}$, while $\mathscr{D}_\Sigma = \mathbb{N}^k$. In what follows we will use the index X_i to refer to the variable X_{A_i}, i.e. we associate an index i to each agent identifier in Σ. At the same time we will use i_A, or sometimes just the agent name A, to refer to the index of agent A. For any system S, we can consider the vector $\mathbf{d}_S \in \mathbb{N}^k$ associating with each variable X_i the value $S[A_i]$. Similarly, for any $\mathbf{d} \in \mathbb{N}^k$ we can consider the system $S_\mathbf{d}$ such that $S[A_i] = \mathbf{d}[i]$. The initial state $\mathbf{d}_\Sigma$ of the model M_Σ is equal to $\mathbf{d}_S$.

The set of transitions $\mathscr{T}_\Sigma$ in M_Σ consists of two groups of rules modelling *unicast* and *broadcast* interactions. The elements of the first group will be denoted by $\tau^{\langle A_i, a, A_j\rangle}$ and models the unicast interaction between an agent A_i that is sending a (unicast) signal on a that is received by A_j.

Example 10. Let us consider the *running example* used in Section 2. We can associate each configuration S of our model with a vector $\mathbf{X}_{ca}$ of *four variables*, each taking values in $\mathbb{N}$. The structure of $\mathbf{X}_{ca}$ is the following:

$$\mathbf{X}_{ca} = (X_{\mathsf{R}}, X_{\mathsf{B}}, X_{\mathsf{RT}}, X_{\mathsf{BT}}).$$

The vector $\mathbf{d}$ associated with the system of Example 3 is:

$$(97 \cdot N, N, N, N)$$

□

Given $\Sigma = \langle S, \varepsilon, \Delta \rangle$, for any $A_i, A_j \in \mathsf{agents}(\Delta, S)$, we let

$$\tau^{\langle i,a,j \rangle} = (a_{(i,j)}, \pi^a_{i,j}, r^a_{i,j}) \tag{5}$$

where:

$$\begin{aligned} \pi^a_{i,j}(\mathbf{d}) &= [-\mathbf{1}_{A_i} \mapsto 1] \otimes [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i} \\ &\otimes [-\mathbf{1}_{A_j} \mapsto 1] \otimes [\mathbf{1}_{B_j} \mapsto \mathbb{P}(A_j, a?, B_j)]_{B_j : A_j \overset{a?}{\mapsto}_\Delta B_j} \end{aligned} \tag{6}$$

$$r^a_{i,j}(\mathbf{d}) = \mathbf{d}[i] \cdot \mathbf{r}^{\varepsilon}_{S_{\mathbf{d}}}(A_i, a!) \cdot \frac{(\mathbf{d} - \mathbf{1}_i)[j] \cdot \mathbf{w}^{\varepsilon}_{S_{\mathbf{d}}}(A_j, a?)}{\sum_{l : A_l \overset{a?}{\mapsto}_\Delta} (\mathbf{d} - \mathbf{1}_i)[l] \cdot \mathbf{w}^{\varepsilon}_{S_{\mathbf{d}}}(A_l, a?)} \tag{7}$$

Above, we use $\mathbf{1}_{A_i}$ to denote the vector $\mathbf{d} \in \mathbb{N}^k$ having $\mathbf{d}[i] = 1$ and 0 on all the other indices. Moreover, for each $\pi_1, \pi_2 \in Dist_F(\mathbb{R}^n)$, $\pi_1 \otimes \pi_2$ denotes the probability distribution π in $Dist_F(\mathbb{R}^n)$ such that:

$$\pi(\mathbf{v}) = \sum_{\mathbf{v}_1 \in dom(\pi_1)} \pi_1(\mathbf{v}_1) \cdot \sum_{\mathbf{v}_2 \in dom(\pi_2) : \mathbf{v}_1 + \mathbf{v}_2 = \mathbf{v}} \pi_2(\mathbf{v}_2)$$

A rule $\tau^{\langle i,a,j \rangle}$ states that in the update associated with a unicast output performed by agent A_i that is received by A_j, an instance of A_i is removed with probability 1 ($[-\mathbf{1}_{A_i} \mapsto 1]$) while one of the agents B_i reachable from A_i after a $a!$ action is activated with probability $\mathbb{P}(A_i, a!)(B_i)$ (denoted by $[\mathbf{1}_{B_i} \mapsto \mathbb{P}_\varepsilon(A_i, a!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i}$). At the same time, one instance of A_j is removed with probability 1 while one of the agents B_j reachable from A_j after a $a?$ action is activated with probability $\mathbb{P}(A_j, a?)(B_j)$ ($[\mathbf{1}_{B_j} \mapsto \mathbb{P}_\varepsilon(A_j, a?, B_j)]_{B_j : A_j \overset{a?}{\mapsto}_\Delta B_j}$). This update is executed with a rate that is the rate of a in agent A_i ($\mathbf{r}(A_i, a)$) multiplied by the number of instances of A_i (that is $\mathbf{d}[i]$) and by the probability that one instance of A_j receives the signal ($\frac{(\mathbf{d} - \mathbf{1}_i)[j] \cdot \mathbf{w}(A_j, a?)}{\sum_l (\mathbf{d} - \mathbf{1}_i)[l] \cdot \mathbf{w}(A_l, a?)}$).

Example 11. Let us consider rule $\tau^{\mathsf{BT,beRed,B}}$ describing the interaction of an agent in state BT with one in B via a unicast synchronisation on action beRed. We have that:

$$\begin{aligned} \pi^{\mathsf{beRed}}_{\mathsf{BT,B}}(\mathbf{d}) &= [-\mathbf{1}_{\mathsf{BT}} \mapsto 1] \otimes [\mathbf{1}_{\mathsf{B}} \mapsto 1] \otimes [-\mathbf{1}_{\mathsf{B}} \mapsto 1] \otimes [\mathbf{1}_{\mathsf{R}} \mapsto 1] \\ &= [-\mathbf{1}_{\mathsf{BT}} + \mathbf{1}_{\mathsf{B}} - \mathbf{1}_{\mathsf{B}} + \mathbf{1}_{\mathsf{R}} \mapsto 1] \\ &= [\mathbf{1}_{\mathsf{R}} - \mathbf{1}_{\mathsf{BT}} \mapsto 1] \end{aligned}$$

Hence, when rule $\tau^{\mathsf{BT},\mathsf{beRed},\mathsf{B}}$ is applied only one update can occur with probability 1: $\mathbf{1}_{\mathsf{R}} - \mathbf{1}_{\mathsf{BT}}$.

The rate $r^{\mathsf{beRed}}_{\mathsf{BT},\mathsf{B}}(\mathbf{d})$ of the rule is the following:

$$\begin{aligned}
r^{\mathsf{beRed}}_{\mathsf{BT},\mathsf{B}}(\mathbf{d}) &= \mathbf{d}[\mathsf{BT}] \cdot \mathbf{r}^{\varepsilon_{ca}}_{S_{\mathbf{d}}}(\mathsf{BT},\mathsf{beRed}!) \cdot \frac{(\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[\mathsf{B}] \cdot \mathbf{w}^{\varepsilon_{ac}}_{S_{\mathbf{d}}}(\mathsf{B},\mathsf{beRed}?)}{\sum_{A:A \overset{\mathsf{beRed}?}{\mapsto}_{\Delta_{ac}}} (\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[A] \cdot \mathbf{w}^{\varepsilon_{ac}}_{S_{\mathbf{d}}}(A,\mathsf{beRed}?)} \\
&= \mathbf{d}[\mathsf{BT}] \cdot \mathbf{r}^{\varepsilon_{ca}}_{S_{\mathbf{d}}}(\mathsf{BT},\mathsf{beRed}!) \cdot \frac{(\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[\mathsf{B}]}{\sum_{A:A \overset{\mathsf{beRed}?}{\mapsto}_{\Delta_{ac}}} (\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[A]} \\
&= \mathbf{d}[\mathsf{BT}] \cdot \mathbf{r}^{\varepsilon_{ca}}_{S_{\mathbf{d}}}(\mathsf{BT},\mathsf{beRed}!) \cdot \frac{(\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[\mathsf{B}]}{(\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[\mathsf{B}] + (\mathbf{d}-\mathbf{1}_{\mathsf{BT}})[\mathsf{BT}]} \\
&= \mathbf{d}[\mathsf{BT}] \cdot \lambda_c \cdot \frac{\mathbf{d}[\mathsf{B}]}{\mathbf{d}[\mathsf{B}] + \mathbf{d}[\mathsf{BT}] - 1}
\end{aligned}$$

□

The group of *broadcast* rules contains elements $\tau^{\langle A_i, a^\star \rangle}$ regulating possible behaviours induced by the execution of a *broadcast output* on action a by an agent A.

Let $\Sigma = \langle S, \varepsilon, \Delta \rangle$, for any $A_i \in \mathsf{agents}(\Delta, S)$, we let $\tau^{\langle i, a^\star \rangle} = (a_i^\star, \pi_i^{a^\star}, r_i^{a^\star})$. Function $\pi_i^{a^\star}$, that describes the update induced by the action, is defined as follows:

$$\begin{aligned}
\pi_i^{a^\star}(\mathbf{d}) &= [-\mathbf{1}_{A_i} \mapsto 1] \otimes [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a^\star!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i} \\
&\quad \otimes \bigotimes_{A_j : A_j \overset{a^\star?}{\mapsto}_\Delta} \mathbf{u}_j^{a^\star}((\mathbf{d}-\mathbf{1}_{A_i})(j), \mathbf{p}^{\varepsilon}_{S_{\mathbf{d}}}(A_j, a^\star?))
\end{aligned} \tag{8}$$

where:

$$\mathbf{u}_j^{a^\star}(n,p) = \sum_{k=0}^{n} \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} [-\mathbf{1}_{A_j} \mapsto 1]^k \otimes [\mathbf{1}_{B_j} \mapsto \mathbb{P}(A_j, a^\star?, B_j)]^k_{B_j : A_j \overset{a^\star?}{\mapsto}_\Delta B_j}$$

while the rate function $r_i^{a^\star}$ is:

$$r_i^{a^\star}(\mathbf{d}) = \mathbf{d}[i] \cdot \mathbf{r}^{\varepsilon}_{S_{\mathbf{d}}}(A_i, a^\star!) \tag{9}$$

The update function $\pi_i^{a^\star}$ states that when the rule is applied one instance of A_i is replaced by one of the agents B_i such that $A_i \overset{a^\star!}{\mapsto} B_i$. The resulting update is governed by the probability distribution $\mathbb{P}(A_i, a^\star!)$. Any agent A_j that can perform the complementary broadcast input, i.e. $A_j \overset{a^\star?}{\mapsto}_\Delta$, is potentially involved in the synchronisation. The number of instances involved in the update depends on the probability $\mathbf{p}^{\varepsilon}_{S_{\mathbf{d}}}(A_j, a^\star?)$. The function $\mathbf{u}_j^{a^\star}(n,p)$ is used to compute the probability distribution of possible updates induced on n copies of agent A_j that can receive the signal with probability p. This is in fact a *multinomial distribution*: each of the n copies receives the message with probability p and evolves to a new agent selected according to the probability distribution $\mathbb{P}(A_j, a^\star?)$. Given a probability distribution π, we let π^k be the $\chi_{\mathbf{0}}$ distribution when $k = 0$, and $\pi \otimes \pi^{k-1}$ when $k > 0$. The rate of the update $r_i^{a^\star}(\mathbf{d})$ is just the rate of $a^\star!$ in A_i multiplied by the number of instances of $\mathbf{d}[i]$ of A_i.

Example 12. In our running example (see Example 4), agent R uses action $\mathsf{red}^\star!$ to *advertise* its colour. These interactions are rendered via rule $\tau^{\mathsf{R},\mathsf{red}^\star} = (\mathsf{red}^\star, \pi_{\mathsf{R}}^{\mathsf{red}^\star}, r_{\mathsf{R}}^{\mathsf{red}^\star})$ where:

$$\begin{aligned}
\pi_{\mathsf{R}}^{\mathsf{red}^\star}(\mathbf{d}) &= [-\mathbf{1}_{\mathsf{R}} \mapsto 1] \otimes [\mathbf{1}_{\mathsf{R}} \mapsto 1] \\
&\quad \otimes\ \mathbf{u}_{\mathsf{R}}^{\mathsf{red}^\star}(\mathbf{d}[\mathsf{R}] - 1, \mathbf{p}_{S_{\mathbf{d}}}^{\varepsilon_{ac}}(\mathsf{R}, \mathsf{red}^\star?)) \\
&\quad \otimes\ \mathbf{u}_{\mathsf{BT}}^{\mathsf{red}^\star}(\mathbf{d}[\mathsf{BT}], \mathbf{p}_{S_{\mathbf{d}}}^{\varepsilon_{ac}}(\mathsf{BT}, \mathsf{red}^\star?)) \\
&= [-\mathbf{1}_{\mathsf{R}} \mapsto 1] \otimes [\mathbf{1}_{\mathsf{R}} \mapsto 1] \\
&\quad \otimes\ \mathbf{u}_{\mathsf{R}}^{\mathsf{red}^\star}\left(\mathbf{d}[\mathsf{R}] - 1, \frac{\kappa}{\mathbf{d}[\mathsf{R}] + \mathbf{d}[\mathsf{BT}]}\right) \\
&\quad \otimes\ \mathbf{u}_{\mathsf{BT}}^{\mathsf{red}^\star}\left(\mathbf{d}[\mathsf{BT}], \frac{\kappa}{\mathbf{d}[\mathsf{R}] + \mathbf{d}[\mathsf{BT}]}\right)
\end{aligned}$$

and

$$r_{\mathsf{R}}^{\mathsf{red}^\star}(\mathbf{d}) \;=\; \mathbf{d}[\mathsf{R}] \cdot \mathbf{r}_{S_{\mathbf{d}}}^{\varepsilon_{ac}}(\mathsf{R}, \mathsf{red}^\star!) \;=\; \mathbf{d}[\mathsf{R}] \cdot \lambda_a$$

The update induced by rule $\tau^{\mathsf{R},\mathsf{red}^\star}$ consists of the combination of the multinomials distributions computed by functions $\mathbf{u}_{\mathsf{R}}^{\mathsf{red}^\star}$ and $\mathbf{u}_{\mathsf{BT}}^{\mathsf{red}^\star}$:

$$\begin{aligned}
\mathbf{u}_{\mathsf{R}}^{\mathsf{red}^\star}(n,p) &= \textstyle\sum_{k=0}^{n} \binom{n}{k}(1-p)^{n-k} \cdot p^k \\
&\quad \cdot \textstyle\sum_{k_1+k_2=k} \binom{k}{k_1\ k_2} [k_2 \cdot (\mathbf{1}_{\mathbf{RT}} - \mathbf{1}_{\mathbf{R}}) \mapsto (1-p_t)^{k_1} \cdot p_t^{k_2}] \\
\mathbf{u}_{\mathsf{BT}}^{\mathsf{red}^\star}(n,p) &= \textstyle\sum_{k=0}^{n} \binom{n}{k}(1-p)^{n-k} \cdot p^k \\
&\quad \cdot \textstyle\sum_{k_1+k_2=k} \binom{k}{k_1\ k_2} [k_2 \cdot (\mathbf{1}_{\mathbf{B}} - \mathbf{1}_{\mathbf{BT}}) \mapsto (1-p_c)^{k_1} \cdot p_c^{k_2}]
\end{aligned}$$

□

Definition 4. Let $\Sigma = \langle S, \varepsilon, \Delta \rangle$ be a NBA specification, then the associated population model of Σ is $\mathbf{M}_\Sigma = (\mathbf{X}_\Sigma, \mathscr{D}_\Sigma, \mathscr{T}_\Sigma, \mathbf{d}_\Sigma)$ such that:

- $\mathbf{X}_\Sigma = \{A_i, \ldots, A_k\} = \textsc{Ag}[\Sigma]$;
- $\mathscr{D}_\Sigma = \mathbb{N}^k$;
- $\mathscr{T}_\Sigma = \{\tau^{\langle i,a,j \rangle} \mid A_i \overset{a!}{\mapsto}_\Delta \text{ and } A_j \overset{a?}{\mapsto}_\Delta\} \cup \{\tau^{\langle i,a^\star \rangle} \mid A_i \overset{a^\star!}{\mapsto}_\Delta\}$;
- $d_\Sigma = d_S$.

The following theorem guarantees that the CTMC associated with a specification Σ and the one associated with the *population model* $\mathbf{M}_\Sigma$ identify the same stochastic process.

Theorem 6. *Let* $\Sigma = \langle S, \varepsilon, \Delta \rangle$ *and* $\mathbf{M}_\Sigma = (\mathbf{X}_\Sigma, \mathscr{D}_\Sigma, \mathscr{T}_\Sigma, \mathbf{d}_S)$. *The CTMCs* $(\textsc{Sys}, \mathbf{Q}_\Sigma)$ *and* $(\mathbb{N}^k, \mathbf{Q}_{\mathbf{M}_\Sigma})$ *represent the same* stochastic process.

PROOF. The proof is based on the fact that for any $S_1, S_2 \in \text{SYS}$ and for any $\mathbf{d}_1, \mathbf{d}_2 \in \mathbb{N}^k$, the following hold:

1. $\mathbf{Q}_\Sigma[S_1, S_2] = \mathbf{Q}_{\mathbf{M}_\Sigma}[\mathbf{d}_{S_1}, \mathbf{d}_{S_2}]$;
2. $\mathbf{Q}_{\mathbf{M}_\Sigma}[\mathbf{d}_1, \mathbf{d}_2] = \mathbf{Q}_\Sigma[S_{\mathbf{d}_1}, S_{\mathbf{d}_2}]$.

We first introduce the auxiliary functions $\mathscr{F}^{\varepsilon,a}_{A_i,A_j}$ and $\mathscr{F}^{\varepsilon,a^\star}_{A_i}$ defined below:

$$\mathscr{F}^{\varepsilon,a}_{A_i,A_j}(S) = (S[A_i] \cdot \mathbf{r}^\varepsilon_S(A_i, a!) \cdot \mathbb{P}(A_i, a!) \parallel \frac{\mathbb{R}^\Delta_{\varepsilon(S)}((S - A_i) \downarrow A_j, a?)}{\sum_A \mathbb{R}^\Delta_{\varepsilon(S)}(S - A_i) \downarrow A, a?)} \parallel (S - A_i) \backslash A_j$$

$$\mathscr{F}^{\varepsilon,a^\star}_{A_i} = (S[A_i] \cdot \mathbf{r}^\varepsilon_S(A_i, a!) \cdot \mathbb{P}(A_i, a^\star!) \parallel_A \mathbb{R}^\Delta_{\varepsilon(S)}((S - A_i) \downarrow A, a^\star?)$$

where $S \downarrow A$ indicates the system containing only the agents A, while $S \backslash A$ is the one obtained from S by removing all the instances of A.

It is easy to see that:

1. $\mathbb{S}^\Delta_\varepsilon(S, a) = \sum_{A_i} \sum_{A_j} \mathscr{F}^{\varepsilon,a}_{A_i,A_j}(S)$;
2. $\mathbb{S}^\Delta_\varepsilon(S, a^\star) = \sum_{A_i} \mathscr{F}^{\varepsilon,a^\star}_{A_i}(S)$;
3. $\mathscr{F}^{\varepsilon,a}_{A_i,A_j}(S_1, S_2) = v$ if and only if $v = r^a_{i,j}(\mathbf{d}_{S_1}) \cdot \pi^a_{i,j}(\mathbf{d}_{S_2} - \mathbf{d}_{S_1})$;
4. $\mathscr{F}^{\varepsilon,a^\star}_{A_i}(S_1, S_2) = v$ if and only if $v = r^{a^\star}_i(\mathbf{d}_{S_1}) \cdot \pi^{a^\star}_i(\mathbf{d}_{S_2} - \mathbf{d}_{S_1})$;

Directly from the four items above and from the definition of $\mathbf{Q}_\Sigma$ and $\mathbf{Q}_{\mathbf{M}_\Sigma}$ we have that for any $S_1, S_2 \in \text{SYS}$ and for any $\mathbf{d}_1, \mathbf{d}_2 \in \mathbb{N}^k$, the following two equalities hold:

1. $\mathbf{Q}_\Sigma[S_1, S_2] = \mathbf{Q}_{\mathbf{M}_\Sigma}[\mathbf{d}_{S_1}, \mathbf{d}_{S_2}]$;
2. $\mathbf{Q}_{\mathbf{M}_\Sigma}[\mathbf{d}_1, \mathbf{d}_2] = \mathbf{Q}_\Sigma[S_{\mathbf{d}_1}, S_{\mathbf{d}_2}]$.

□

Thanks to Theorem 6 we are guaranteed that any NBA specification Σ can be *translated* into an equivalent *population model* $\mathbf{M}_\Sigma$. This facilitates the use of the technique presented in Section 3 that enables *scalable analysis* of NBA models. However, we have to guarantee that all the conditions of Definition 3 are satisfied.

Lemma 7. *Let* $\Sigma = \langle S, \varepsilon, \Delta \rangle$ *and* $\mathbf{M}_\Sigma = (\mathbf{X}_\Sigma, \mathscr{D}_\Sigma, \mathscr{T}_\Sigma, \mathbf{d}_S)$. *The following hold:*

- *for any* $\tau^{\langle i,a,j \rangle} \in \mathscr{T}_\Sigma$

$$\mu_{\tau^{\langle i,a,j \rangle}}(\mathbf{d}) \quad = \quad -\mathbf{1}_{A_i} + \sum_{B_i} \mathbb{P}(A_i, a!, B_i) \cdot \mathbf{1}_{B_i} - \mathbf{1}_{A_j} + \sum_{B_j} \mathbb{P}(A_j, a?, B_j) \cdot \mathbf{1}_{B_j}$$

- *for any* $\tau^{\langle i,a^\star\rangle} \in \mathscr{T}_\Sigma$

$$\begin{array}{rcl}\mu_{\tau^{\langle i,a^\star\rangle}}(\mathbf{d}) & = & -\mathbf{1}_{A_i} + \sum_{B_i} \mathbb{P}(A_i, a^\star!, B_i) \cdot \mathbf{1}_{B_i} \\ & + & \sum_{A_j : A_j \overset{a^\star?}{\mapsto}_\Delta} \mathbf{p}^{\varepsilon}_{S_\mathbf{d}}(A_j, a^\star) \cdot (\mathbf{d} - \mathbf{1}_{A_i})(j) \cdot \left(-\mathbf{1}_{A_j} + \sum_{B_j} \mathbb{P}(A_j, a^\star?, B_j) \cdot \mathbf{1}_{B_j} \right)\end{array}$$

PROOF. First of all we can observe that for any $\pi_1, \pi_2 \in \mathit{Dist}(\mathscr{D})$ we have that:

$$\sum_{\mathbf{v} \in \mathscr{D}} \mathbf{v} \cdot (\pi_1 \otimes \pi_2)(\mathbf{v}) = \sum_{\mathbf{v_1} \in \mathscr{D}} \mathbf{v_1} \cdot \pi_1(\mathbf{v_1}) + \sum_{\mathbf{v_2} \in \mathscr{D}} \mathbf{v_2} \cdot \pi_2(\mathbf{v_2}) \tag{10}$$

Indeed:

$$\begin{array}{rcl}\sum_{\mathbf{v} \in \mathscr{D}} \mathbf{v} \cdot (\pi_1 \otimes \pi_2)(\mathbf{v}) & = & \sum_{\mathbf{v} \in \mathscr{D}} \mathbf{v} \cdot \left(\sum_{\mathbf{v_1} \in \mathscr{D}} \pi_1(\mathbf{v_1}) \cdot \sum_{\mathbf{v_2} \in \mathscr{D} | \mathbf{v} = \mathbf{v_1} + \mathbf{v_2}} \pi_2(\mathbf{v_2}) \right) \\ & = & \sum_{\mathbf{v} \in \mathscr{D}} \sum_{\mathbf{v_1} \in \mathscr{D}} \sum_{\mathbf{v_2} \in \mathscr{D} | \mathbf{v} = \mathbf{v_1} + \mathbf{v_2}} \mathbf{v} \cdot (\pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v_2})) \\ & = & \sum_{\mathbf{v_1} \in \mathscr{D}} \sum_{\mathbf{v_2} \in \mathscr{D}} (\mathbf{v_1} + \mathbf{v_2}) \cdot (\pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v_2})) \\ & = & \sum_{\mathbf{v_1} \in \mathscr{D}} \sum_{\mathbf{v_2} \in \mathscr{D}} (\mathbf{v_1} \cdot (\pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v})) + \mathbf{v_2} \cdot (\pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v_2}))) \\ & = & \sum_{\mathbf{v_1} \in \mathscr{D}} \sum_{\mathbf{v_2} \in \mathscr{D}} \mathbf{v_1} \cdot \pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v_2}) \\ & & + \sum_{\mathbf{v_1} \in \mathscr{D}} \sum_{\mathbf{v_2} \in \mathscr{D}} \mathbf{v_2} \pi_1(\mathbf{v_1}) \cdot \pi_2(\mathbf{v_2})) \\ & = & \sum_{\mathbf{v_1} \in \mathscr{D}} \mathbf{v_1} \cdot \pi_1(\mathbf{v_1}) \cdot \sum_{\mathbf{v_2} \in \mathscr{D}} \pi_2(\mathbf{v_2}) \\ & & + \sum_{\mathbf{v_2} \in \mathscr{D}} \mathbf{v_2} \cdot \pi_2(\mathbf{v}) \cdot \sum_{\mathbf{v_1} \in \mathscr{D}} \pi_1(\mathbf{v_1}) \\ & = & \sum_{\mathbf{v_1} \in \mathscr{D}} \mathbf{v_1} \cdot \pi_1(\mathbf{v_1}) + \sum_{\mathbf{v_2} \in \mathscr{D}} \mathbf{v_2} \cdot \pi_2(\mathbf{v_2})\end{array}$$

We have that $\tau^{\langle i,a,j\rangle} = (a_{(i,j)}, \pi^a_{i,j}, r^a_{i,j})$, where:

$$\begin{array}{rcl}\pi^a_{i,j}(\mathbf{d}) & = & [-\mathbf{1}_{A_i} \mapsto 1] \otimes [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i} \\ & \otimes & [-\mathbf{1}_{A_j} \mapsto 1] \otimes [\mathbf{1}_{B_j} \mapsto \mathbb{P}(A_j, a?, B_j)]_{B_j : A_j \overset{a?}{\mapsto}_\Delta B_j}\end{array}$$

Directly from Equation 10 we have that:

$$\begin{array}{rcl}\mu_{\tau^{\langle i,a,j\rangle}}(\mathbf{d}) = \sum_{\mathbf{v} \in \mathscr{D}} \pi^a_{i,j}(\mathbf{d})(\mathbf{v}) & = & \sum_{\mathbf{v} \in \mathscr{D}} [-\mathbf{1}_{A_i} \mapsto 1](\mathbf{v}) \\ & + & \sum_{\mathbf{v} \in \mathscr{D}} [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i}(\mathbf{v}) \\ & + & \sum_{\mathbf{v} \in \mathscr{D}} [-\mathbf{1}_{A_j} \mapsto 1](\mathbf{v}) \\ & + & \sum_{\mathbf{v} \in \mathscr{D}} [\mathbf{1}_{B_j} \mapsto \mathbb{P}(A_j, a?, B_j)]_{B_j : A_j \overset{a?}{\mapsto}_\Delta B_j}(\mathbf{v})\end{array}$$

Finally, by observing that

- $\sum_{\mathbf{v} \in \mathscr{D}} [-\mathbf{1}_{A_i} \mapsto 1](\mathbf{v}) = -\mathbf{1}_{A_i}$
- $\sum_{\mathbf{v} \in \mathscr{D}} [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i}(\mathbf{v}) = \sum_{B_i} \mathbb{P}(A_i, a!, B_i) \cdot \mathbf{1}_{B_i}$

We can conclude that:

$$\mu_{\tau^{\langle i,a,j\rangle}}(\mathbf{d}) = -\mathbf{1}_{A_i} + \textstyle\sum_{B_i} \mathbb{P}(A_i, a!, B_i) \cdot \mathbf{1}_{B_i} - \mathbf{1}_{A_j} + \sum_{B_j} \mathbb{P}(A_j, a?, B_j) \cdot \mathbf{1}_{B_j}$$

The proof of the second item in the statement follows analogously by simple algebra and by observing that $\tau^{\langle i,a^\star\rangle} = (a_i^\star, \pi_i^{a^\star}, r_i^{a^\star})$, where:

$$\begin{aligned}\pi_i^{a^\star}(\mathbf{d}) &= [-\mathbf{1}_{A_i} \mapsto 1] \otimes [\mathbf{1}_{B_i} \mapsto \mathbb{P}(A_i, a^\star!, B_i)]_{B_i : A_i \overset{a!}{\mapsto}_\Delta B_i} \\ &\quad \otimes \bigotimes_{A_j : A_j \overset{a^\star?}{\mapsto}_\Delta} \mathbf{u}_j^{a^\star}((\mathbf{d} - \mathbf{1}_{A_i})(j), \mathbf{p}^{\varepsilon}_{S_\mathbf{d}}(A_j, a^\star?))\end{aligned}$$

and

$$\mathbf{u}_j^{a^\star}(n,p) = \sum_{k=0}^{n} \binom{n}{k} \cdot p^k \cdot (1-p)^{n-k} [-\mathbf{1}_{A_j} \mapsto 1]^k \otimes [\mathbf{1}_{B_j} \mapsto \mathbb{P}(A_j, a^\star?, B_j)]^k_{B_j : A_j \overset{a^\star?}{\mapsto}_\Delta B_j}$$

Indeed, $\mathbf{u}_j^{a^\star}(n,p)$ is in fact a *multinomial distribution* and its mean is:

$$p \cdot n \cdot \left(-\mathbf{1}_{A_j} + \sum_{B_j} \mathbb{P}(A_j, a^\star?, B_j) \cdot \mathbf{1}_{B_j} \right)$$

□

Finally, the following theorem guarantees that if Σ is a *scale invariant* specification, the population model M_Σ satisfies the conditions of Definition 3. Hence, we can approximate the stochastic behaviour of Σ via a system of ODE.

Theorem 8. *Let* $\Sigma = \langle S, \varepsilon, \Delta \rangle$ *be* scale invariant. *The population model* $M_\Sigma = (\mathbf{X}_\Sigma, \mathscr{D}_\Sigma, \mathscr{T}_\Sigma, \mathbf{d}_S)$ *satisfies all the conditions of Definition 3.*

PROOF. We consider a sequence of NBA specifications $\{\Sigma_k\}_{k \in \mathbb{N}_{>0}}$, where $\Sigma_k = \langle k \cdot S, \varepsilon, \Delta \rangle$, and the corresponding sequence of population models $(M_{\Sigma_k})_{k \in \mathbb{N}_{>0}}$. We can observe that the *normalised* PCTMC model $(\hat{M}^N)_{N > |S|}$ *admits a fluid approximation*. Indeed:

- the system size grows linearly with $N = k \cdot |S|$;
- there exists a subset $E \subseteq \mathbb{R}^n$ such that $\hat{\mathscr{D}}^N \subseteq E$ for all N, this set is $\{\frac{x}{N} \mid x \in \mathbb{N}\}$;
- for each $\tau \in \mathscr{T}^N$ there exists a locally Lipschitz continuous and bounded function $\mathbf{m}_\tau : E \to \mathbb{R}$ such that uniformly for $\hat{\mathbf{b}} \in E$, $\lim_{N\to\infty} \mu_\tau^N(\frac{1}{\delta_N}\hat{\mathbf{d}}) = \mathbf{m}_\tau(\hat{\mathbf{d}})$. Since ε is *scale invariant*, this function $\mathbf{m}_\tau$ can be obtained from functions $\mu_{\tau_{i,j}^a}$ and $\mu_{\tau_{i,j}^a}$ of Lemma 7 by replacing $\mathbf{p}^{\varepsilon}_{S_\mathbf{d}}(A_j, a^\star) \cdot (\mathbf{d} - \mathbf{1}_{A_i})$ with the average number of agents involved in the broadcast. We can observe that this value is constant and does not change when N increases.
- for each $\tau \in \mathscr{T}^N$, $\sup_N \mathbf{w}_\tau^N \left(\frac{1}{\delta_N} \cdot \hat{\mathbf{d}} \right)$ is locally bounded in E. In the case of rules of the form $\tau_{i,j}^a$, which model unicast interactions, $\mathbf{w}_\tau^N$ does not change while N increases (and goes to 0 in the normalised model). When we consider the rules modelling broadcast interactions, i.e. ones of the form $\tau_i^\star$, we have that $\mathbf{w}_\tau^N$ is bounded by the the average number of agents (that is a constant) involved in a broadcast interaction.

- there is a locally Lipschitz continuous and bounded function $g : E \to \mathbb{R}_{\geq 0}$ such that:

$$\lim_{N\to\infty} \delta_N \hat{r}^N(\hat{\mathbf{d}}) = g(\hat{\mathbf{d}})$$

uniformly for $\hat{\mathbf{d}} \in E$. This function g is obtained directly $\mathbf{r}^{\varepsilon}_{S_{\mathbf{d}}}$. Since ε is scale invariant, this function does not change with N.

□

Example 13. Any NBA specification that is *scale invariant* can be translated into a system of Ordinary Differential Equations (ODE) that approximates its stochastic behaviour. This is the case of the NBA model of Example 3.

The results are reported in Figure 4 where the results of the fluid approximation (obtained by solving the ODE system with SciPy) are compared with the ones obtained via simulation[4]. We can observe that, when N increases, the distance between the two decreases.

□

5. Case Study: Gossip shuffle protocol

In this section we will consider two variants of the *gossip shuffle protocol* (GSP) [24, 2, 3]. In GSP we have a set of agents (or nodes) that share some data items. Due to storage limitations, each agent is able to maintain only a finite list of data items in a local cache. To guarantee that each data item will be eventually available at every node, nodes exchange items by performing *shuffle*: local data are sent to another peer that, symmetrically, sends back its local cache. Following an approach similar to that used in [29], we show how NBA can be used to model the dissemination of a generic data item d in a *network of agents*. According to [24], in GSP each node can be either *active* or *passive*. When an agent is *active* it selects a *passive* participant in the system and sends it a copy of the local cache. The contacted *passive* node replies with the data items stored in its local cache. The participants replace the sent element with the ones received in the interaction. After that, the *active* agent becomes *passive*. *Passive* nodes will become *active* after a randomly selected amount of time.

To model the dissemination of a generic data item d in the system we can consider six different agents: AI, AU, AS, PI, PU and PS. Agent AI models an *active* node that is *informed*, i.e. that has the *data item d* in the local cache. Agents AU and AS identify *active* nodes that are *uninformed*. However, AU represents a node that has never received d, while AS is a node that stored d in the past. Similarly, PI is a *passive* agent that is *informed* while PU and PS identify *passive* and *uninformed* agents, where PS stored d in the past.

Each configuration of GSP can be represented as:

$$\mathsf{AI}[k_1] \parallel \mathsf{AU}[k_2] \parallel \mathsf{AS}[k_3] \parallel \mathsf{PI}[k_4] \parallel \mathsf{PU}[k_5] \parallel \mathsf{PS}[k_6]$$

[4]SciPy script and simulation code are available at http://bit.ly/2R2qt1E.

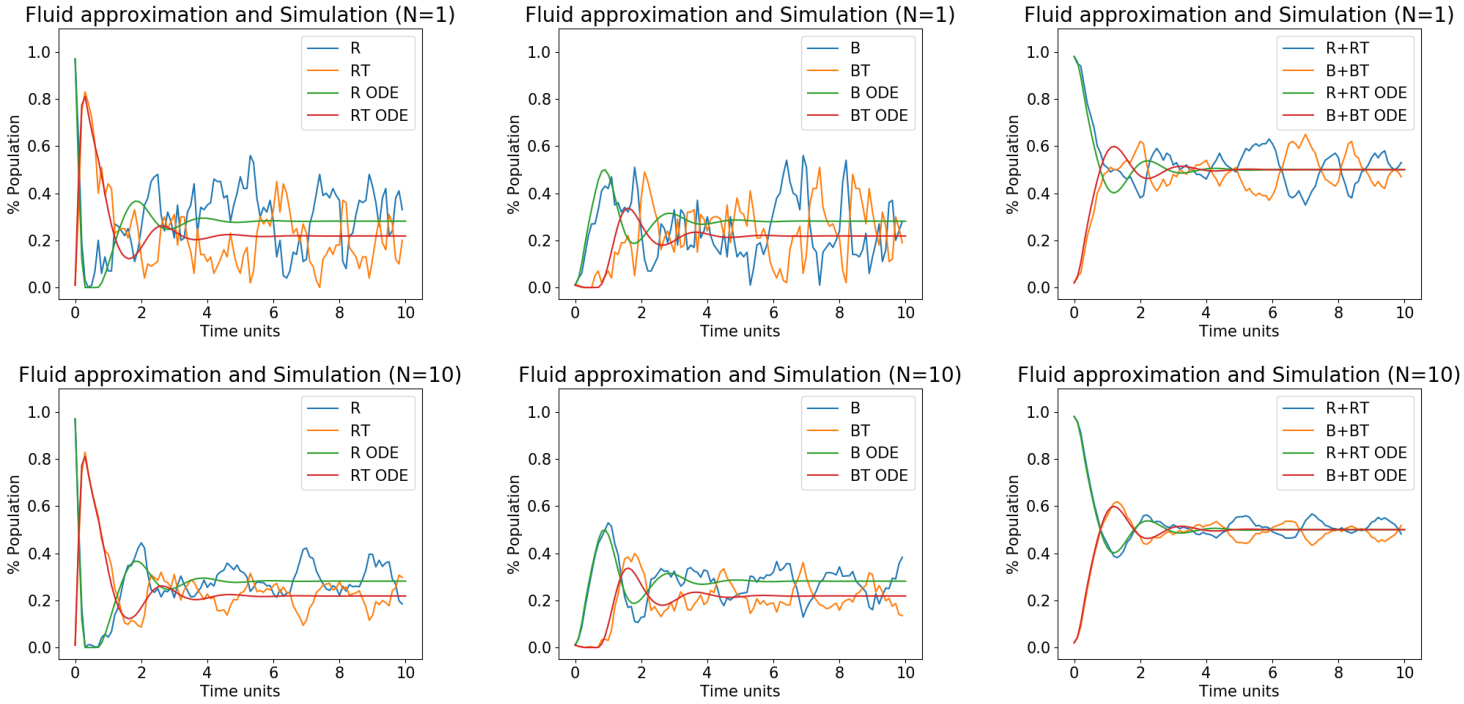

Figure 3: Fluid approximation and Simulation results (single run) of the system of Example 3 showing the fractions of agents in state R and RT (first column), B, and BT (second column), and the fractions of *red* (R or RT) and *blue* (B or BT) agents (third column) with $N = 1$ and $N = 10$ ($\lambda_a = \lambda_c = 1.0$, $p_t = p_c = 0.5$, $\kappa = 10$)

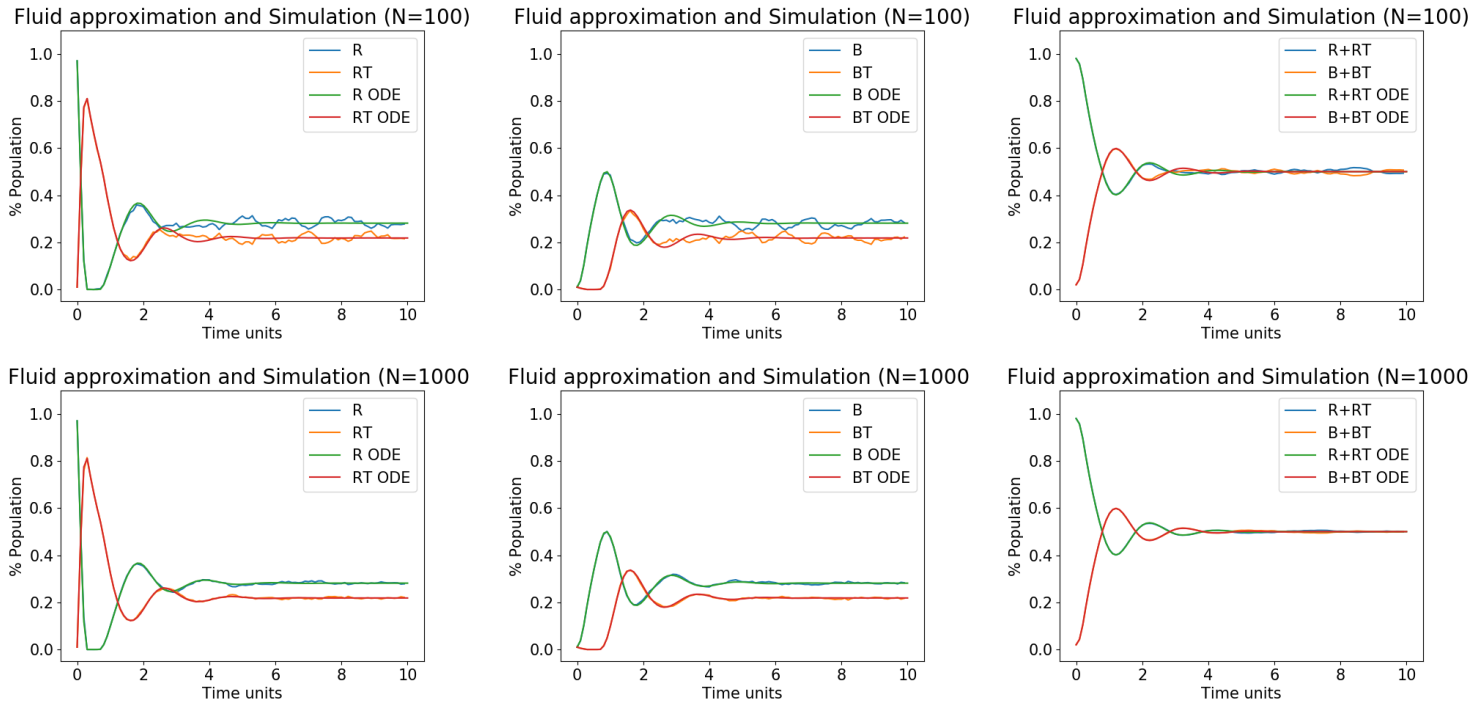

Figure 4: Fluid approximation and Simulation results (single run) of the system of Example 3 showing the fractions of agents in state R and RT (first column), B, and BT (second column), and the fractions of *red* (R or RT) and *blue* (B or BT) agents (third column) with $N = 100$ and $N = 1000$ ($\lambda_a = \lambda_c = 1.0$, $p_t = p_c = 0.5$, $\kappa = 10$)

Interaction between *active* and *passive* agents can be modelled via a unicast synchronisation. *Active* and *informed* agents AI execute output actions of the form $\mathsf{shuffle}^{ii}$ (modelling interactions with *informed passive* agents) and $\mathsf{shuffle}^{iu}$ (modelling interactions with *uninformed passive* agents). For instance, a synchronisation on action $\mathsf{shuffle}^{ii}$ occurs when AI interacts with agent PI, while a synchronisation occurs on $\mathsf{shuffle}^{iu}$ when AI interacts either with an agent PU or with an agent PS. Similarly, an AU (resp. AS) executes actions of the form $\mathsf{shuffle}^{ui}!$ and $\mathsf{shuffle}^{uu}!$ to model the interactions with *informed* and *uninformed* agents, respectively. Passive agents execute a broadcast output $\mathsf{beact}^{\star}!$ to become *active*. The behaviour of agents AI, AU, PI and PU is the following:

$$\begin{aligned}
\mathsf{AI} &\triangleq \mathsf{shuffle}^{ii}!.\mathsf{PI} \oplus \mathsf{shuffle}^{iu}!.\mathsf{PS} \\
\mathsf{AU} &\triangleq \mathsf{shuffle}^{ui}!.\mathsf{PI} \oplus \mathsf{shuffle}^{uu}!.\mathsf{PU} \\
\mathsf{AS} &\triangleq \mathsf{shuffle}^{ui}!.\mathsf{PI} \oplus \mathsf{shuffle}^{uu}!.\mathsf{PS} \\
\mathsf{PI} &\triangleq \mathsf{shuffle}^{ii}?.\mathsf{PI} \oplus \mathsf{shuffle}^{ui}?.\mathsf{PS} \oplus \mathit{beact}^{\star}!.\mathsf{AI} \\
\mathsf{PU} &\triangleq \mathsf{shuffle}^{iu}?.\mathsf{PI} \oplus \mathsf{shuffle}^{uu}?.\mathsf{PU} \oplus \mathit{beact}^{\star}!.\mathsf{AU} \\
\mathsf{PS} &\triangleq \mathsf{shuffle}^{iu}?.\mathsf{PI} \oplus \mathsf{shuffle}^{uu}?.\mathsf{PS} \oplus \mathit{beact}^{\star}!.\mathsf{AS}
\end{aligned}$$

Note that all enabled actions execute with equal probability. Each *active* agent becomes *passive* after an interaction with a *passive* one. An agent that is *informed* becomes *uninformed* when it sends the datum d and the same is not received. Finally, an agent that is *uninformed* becomes *informed* when it receives d.

We have now to define an *environment* ε_{GSP}. Function $\mathbf{r}_S^{\varepsilon_{GSP}}$ (that is defined in Table 3) associates a *rate* with each (output) action that an agent can perform. These values are chosen according to the *duration of the shuffle activity* and the *time an agent remains passive*. We assume that both these values are exponentially distributed *random variables* with rate λ_s and λ_a, respectively. Function $\mathbf{r}_S$ guarantees that the *exit rate* of agents AI, AU, and AS is λ_s, while that of PI, PU and PS is λ_a.

For instance, agent AI executes action $\mathsf{shuffle}^{ii}!$ with rate that is $\lambda_s \cdot \frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]}$. This corresponds to the rate λ_s multiplied by the probability that in the shuffle one of the PI agents is selected. Similarly, agent AU executes action $\mathsf{shuffle}^{ui}!$ with rate $\lambda_s \cdot \frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]}$ while action $\mathsf{shuffle}^{uu}!$ is executed with rate $\lambda_s \cdot \frac{S[\mathsf{PU}]+S[\mathsf{PS}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]}$. In the first case the rate results from the multiplication of λ_s with the probability that one of the PI agents is selected ($\frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]}$). The rate of $\mathsf{shuffle}^{uu}!$ is just obtained by the multiplication of λ_s by the probability that an uninformed agent is selected ($\frac{S[\mathsf{PU}]+S[\mathsf{PS}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S(\mathsf{PS}}$). All the other cases follow a similar pattern.

The other two functions $\mathbf{p}_S^{\varepsilon_{GSP}}$ and $\mathbf{w}_S^{\varepsilon_{GSP}}$ are much simpler and are defined as follows:

$$\mathbf{p}_S(A,\alpha) = 0.0 \qquad \mathbf{w}_S(A,\alpha) = 1.0$$

This means that any agent receives *broadcast messages* with probability 0, that is

$$
\begin{array}{rcl}
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AI}, \mathsf{shuffle}^{ii}!) & = & \lambda_s \cdot \frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AI}, \mathsf{shuffle}^{iu}!) & = & \lambda_s \cdot \frac{S[\mathsf{PU}]+S[\mathsf{PS}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AU}, \mathsf{shuffle}^{ui}!) & = & \lambda_s \cdot \frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AU}, \mathsf{shuffle}^{uu}!) & = & \lambda_s \cdot \frac{S[\mathsf{PU}]+S[\mathsf{PS}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AS}, \mathsf{shuffle}^{ui}!) & = & \lambda_s \cdot \frac{S[\mathsf{PI}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{AS}, \mathsf{shuffle}^{uu}!) & = & \lambda_s \cdot \frac{S[\mathsf{PU}]+S[\mathsf{PS}]}{S[\mathsf{PI}]+S[\mathsf{PU}]+S[\mathsf{PS}]} \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{PI}, \mathsf{beact}^{\star}!) & = & \lambda_a \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{PU}, \mathsf{beact}^{\star}!) & = & \lambda_a \\
\mathbf{r}_S^{\varepsilon_{GSP}}(\mathsf{PS}, \mathsf{beact}^{\star}!) & = & \lambda_a
\end{array}
$$

Table 3: Actions rates of agents AI, AU, PI and PU.

broadcast messages are always ignored by agents. In contrast, *unicast messages* can be received by any agent with the same probability.

We can use the NBA model introduced above to estimate the time needed by an agent to *receive* the *data item d*. Hence, we can contrast the fraction of agents that are *not informed*, i.e. that have never seen the data item d, with the fraction of agents that are or have been *informed*, i.e. that have locally stored d. The first group of agents consists of the agents in state AU or PU. The second group consists of all the other agents.

In Figure 5 the results of simulation and fluid approximation are reported for a population size $N = 10^2$ and $N = 10^3$ starting from a configuration where 80% of the agents are in state PU while the remaining 20% are in state PI. We can observe that the two kinds of analysis almost coincide and that, with the given parameter values, after around 10 time units, 90% of the population has been *informed* about the data item d.

Even if the results of simulation and fluid approximation almost coincide, the computational time needed to obtain these results is quite different. Indeed, while we need around $0.007s$ to compute the solution of the ODE on a standard laptop, a longer time is needed to carry out simulations. In the following table the time needed to complete *a single run of simulation* for different values of the population size N is reported:

N	Simulation Time
10^2	$0.059s$
10^3	$0.167s$
10^4	$1.1729s$
10^5	$10.829s$

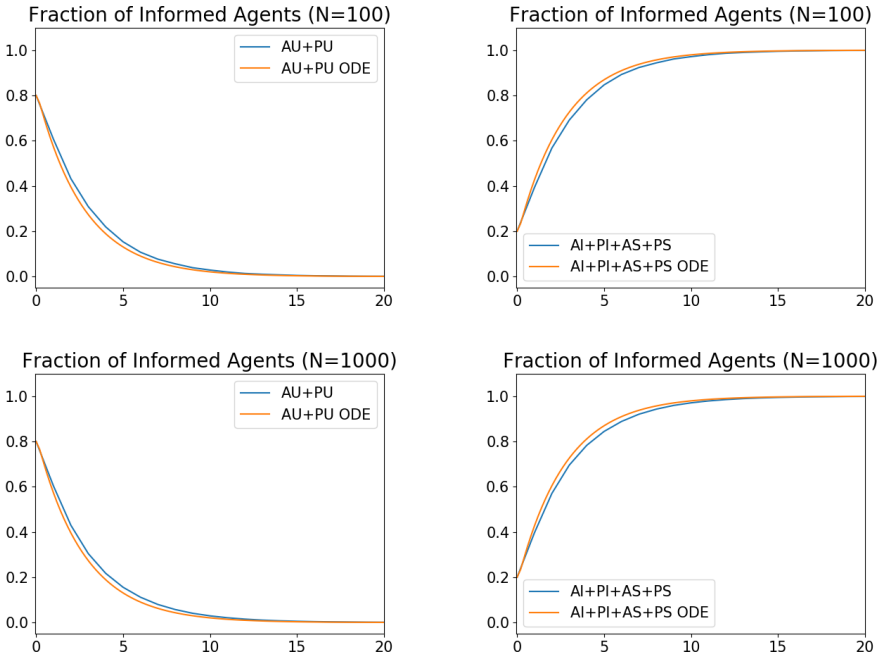

Figure 5: Fluid approximation and simulation (single run) results for GSP ($N = 10^2$ and $N = 10^3$) with $\lambda_s = 10.0$ and $\lambda_a = 1.0$.

5.1. Broadcasting variant

Above, interactions between agents in GSP were rendered in terms of *unicast interaction*. However, in modern system architecture this is not always possible. This is because messages are *spread* in the network and collected by participants that are potentially unknown. In this context one-to-one interactions are often difficult, or even impossible.

For this reason we consider here a variant of the GSP protocol where *active agents* do not interact with a specific participant but they just *broadcast* their data items. The agents that are *passive* can receive these values and use them to update their local cache.

We can consider six different agents: $\mathsf{AI}^\star$, $\mathsf{AU}^\star$, $\mathsf{AS}^\star$, $\mathsf{PI}^\star$, $\mathsf{PU}^\star$ and $\mathsf{PS}^\star$. These identify *active* and *passive* nodes that are either *informed* or *uninformed*, as previously. The behaviour of these agents is defined as follow:

$$
\begin{array}{rcl}
\mathsf{AI}^\star & \triangleq & \mathsf{spread}_i^\star!.\mathsf{PI}^\star \\
\mathsf{AU}^\star & \triangleq & \mathsf{spread}_u^\star!.\mathsf{PU}^\star \\
\mathsf{AS}^\star & \triangleq & \mathsf{spread}_u^\star!.\mathsf{PS}^\star \\
\mathsf{PI}^\star & \triangleq & \mathsf{spread}_i^\star?.\mathsf{PI}^\star \oplus \mathsf{spread}_u^\star?.\mathsf{PS}^\star \oplus \mathit{beact}^\star!.\mathsf{AI}^\star \\
\mathsf{PU}^\star & \triangleq & \mathsf{spread}_i^\star?.\mathsf{PI}^\star \oplus \mathsf{spread}_u^\star?.\mathsf{PU}^\star \oplus \mathit{beact}^\star!.\mathsf{AU}^\star \\
\mathsf{PS}^\star & \triangleq & \mathsf{spread}_i^\star?.\mathsf{PI}^\star \oplus \mathsf{spread}_u^\star?.\mathsf{PS}^\star \oplus \mathit{beact}^\star!.\mathsf{AS}^\star
\end{array}
$$

An agent $\mathsf{AI}^\star$ executes action $\mathsf{spread}_i^\star!$ and then it becomes *passive*. This action indicates that the data item d is spread on the network. Similarly, agents $\mathsf{AU}^\star$ and $\mathsf{AS}^\star$ execute action $\mathsf{spread}_u^\star!$ to model the fact that data idem different from d are communicated. Passive agents $\mathsf{PI}^\star$, $\mathsf{PU}^\star$ and $\mathsf{PS}^\star$ can collect these messages to lose or acquire the data item d. Moreover, they can execute action $\mathsf{beact}^\star!$ to become active.

To complete the specification, we have to define also a function $\varepsilon^\star_{GSP}$ providing the quantitative aspects of a system S. Functions $\mathbf{r}_S^{\varepsilon^\star_{GSP}}$, $\mathbf{p}_S^{\varepsilon^\star_{GSP}}$ and $\mathbf{w}_S^{\varepsilon^\star_{GSP}}$ are defined below:

$$
\begin{array}{rcl}
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{AI}^\star, \mathsf{spread}_i^\star!) & = & \lambda_s \\
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{AU}^\star, \mathsf{spread}_u^\star!) & = & \lambda_s \\
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{AS}^\star, \mathsf{spread}_u^\star!) & = & \lambda_s \\
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{PI}^\star, \mathsf{beact}^\star!) & = & \lambda_a \\
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{PU}^\star, \mathsf{beact}^\star!) & = & \lambda_a \\
\mathbf{r}_S^{\varepsilon^\star_{GSP}}(\mathsf{PS}^\star, \mathsf{beact}^\star!) & = & \lambda_a
\end{array}
$$

$$
\mathbf{p}_S^{\varepsilon^\star_{GSP}}(\mathsf{PI}^\star, \mathsf{spread}_u^\star?) = \begin{cases} k \cdot \frac{1}{S[\mathsf{PI}^\star]+S[\mathsf{PI}^\star]+S[\mathsf{PS}^\star]} & (S[\mathsf{PI}^\star] > 0) \\ 0 & \text{otherwise} \end{cases}
$$

$$
\mathbf{p}_S^{\varepsilon^\star_{GSP}}(\mathsf{PU}^\star, \mathsf{spread}_i^\star?) = \begin{cases} k \cdot \frac{1}{S[\mathsf{PI}^\star]+S[\mathsf{PU}^\star]+S[\mathsf{PS}^\star]} & (S[\mathsf{PU}^\star] > 0) \\ 0 & \text{otherwise} \end{cases}
$$

$$
\mathbf{p}_S^{\varepsilon^\star_{GSP}}(\mathsf{PU}^\star, \mathsf{spread}_i^\star?) = \begin{cases} k \cdot \frac{1}{S[\mathsf{PI}^\star]+S[\mathsf{PU}^\star]+S[\mathsf{PS}^\star]} & (S[\mathsf{PS}^\star] > 0) \\ 0 & \text{otherwise} \end{cases}
$$

We can observe that agents $\mathsf{AI}^\star$, $\mathsf{AU}^\star$ and $\mathsf{AS}^\star$ spread their values with rate λ_s. Passive agents $\mathsf{PI}^\star$, $\mathsf{PU}^\star$ and $\mathsf{PS}^\star$ can receive these messages with probability $k \cdot \frac{1}{S[\mathsf{PI}^\star]+S[\mathsf{PU}]^\star+S[\mathsf{PS}^\star].}$ In these expressions k is used to represent the *average number* of agents that can receive a *broadcast message*, while $\frac{1}{S[\mathsf{PI}^\star]+S[\mathsf{PU}^\star]+S[\mathsf{PS}^\star]}$ represents the probability that an instance of $\mathsf{PI}^\star$ (resp. $\mathsf{PU}^\star$, and $\mathsf{PS}^\star$) is one of the receivers.

We can use simulation and fluid approximation to estimate the time needed by an agent to *receive* at least a copy of the *data item d*. Similarly to what we have done for the unicast version of GSP, we compare the fraction of agents in states $\mathsf{AU}^\star$ or $\mathsf{PU}^\star$, that are the *uninformed* agents, with the fraction of agents in states $\mathsf{AI}^\star$, $\mathsf{PI}^\star$, $\mathsf{AS}^\star$ and $\mathsf{PS}^\star$, that are the *informed* agents. Also in this case we consider an initial configuration where 80% of agents is in state PU while the remaining 20% is PI.

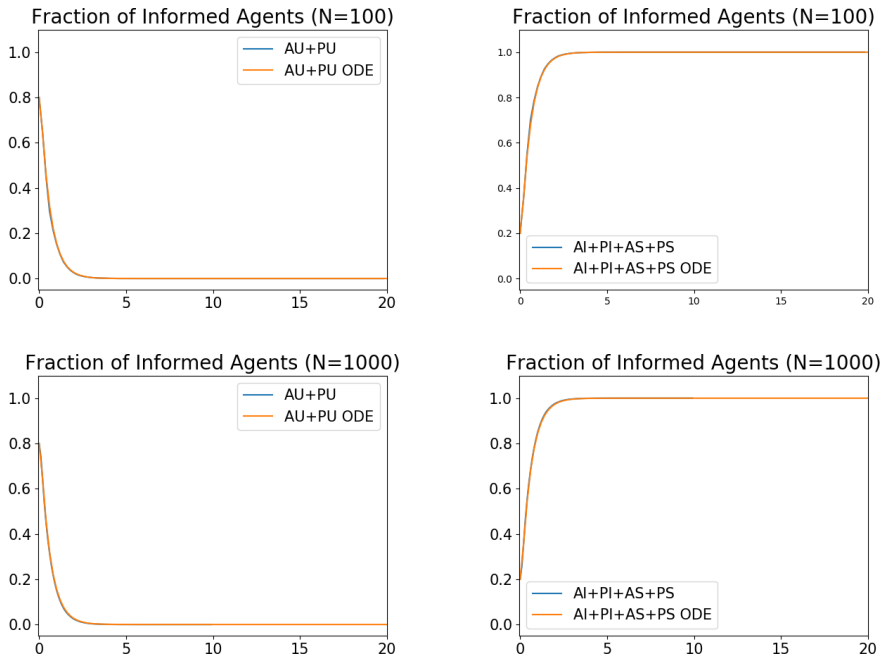

Figure 6: Fluid approximation and simulation results (single run) of broadcast based GSP ($N = 10^2$ and $N = 10^3$) with $\lambda_s = 10.0$, $\lambda_a = 1.0$ and $k = 10$.

In Figure 6 the results of simulation and fluid approximation are reported for population size $N = 10^2$ and $N = 10^3$. We can observe that when broadcast is used, less time is needed than in the unicast mode to spread the data item d. Indeed, after around 2.5 time units 90% of the population has been *informed* about it. Moreover, the initial fraction of agents that are either AI or PI does not change.

The results obtained from the simulation are almost the same as those obtained from the fluid approximation. However, the advantages from a computational point of view are evident. Indeed, only $0.003s$ are needed to compute the solution of the ODE on a standard laptop. While, as reported in the table below, some minutes can be needed to perform *a single run of simulation* when the population size N increases:

N	Simulation Time
10^2	$0.018s$
10^3	$0.206s$
10^4	$15.038s$
10^5	$1481.980s$

The advantages of *fluid approximation* are in this case even more evident. Indeed, to perform a detailed modelling study based on simulation, a large number of runs

(10^2/10^3) must be performed. Days of computations are needed to complete the simulation procedure for large values of N.

6. Concluding Remarks

We have introduced NBA, Network of Broadcasting Agents, a process calculus modelling quantitative aspects of broadcast communication, with the key features of CARMA. This has allowed investigation of the issue of efficient scalable analysis of models incorporating asynchronous, many-to-one communication without the full complexities of CARMA. Thus NBA can be viewed a *proof-of-concept* language which allowed us to explore the issue of fluid approximation of broadcast systems in a focussed way. In this paper we first gave NBA an operational semantics in the style of FuTS, and then provided it with a population semantics, where we count how many agents of each kind are in the system, assuming them indistinguishable. This semantics is used to construct the fluid equations and prove convergence of the NBA stochastic model to this fluid limit under an appropriate scaling of rates and receiving probabilities. The whole calculus is shown at work on two examples, including a Gossip Shuffle Protocol. Thus we have established the feasibility of a fluid approximation for models in a language supporting asynchronous, many-to-one communication. It is worth noting that defining the NBA semantics in terms of population processes opens several alternatives to fluid approximation for efficient analysis, such as moment closures and linear noise approximation [32]. These methods can typically capture stochastic effects that are neglected by the fluid approximation, but at the price of a larger number of differential equations to be solved.

NBA is a simple calculus, but already powerful enough to describe the behaviour of a large class of systems. One way forward, to incorporate these results into a richer language such as CARMA, is to regard NBA as a target language to map more complex and expressive calculi, expanding the attribute-based communication into an appropriate set of communication channels. To fully capture CARMA behaviour, however, we also need to consider dynamically changing environments. This brings extra challenges, as the dynamic of the environment has to be coupled to that of the population model, resulting in a hybrid system rather than a set of differential equations. Hybrid systems can also be obtained as limits of NBA models where broadcast actions can reach a constant fraction of the population, as in the scenario of a national emergency, in which the government sends a text message to all the citizens, sketched after the introduction of the condition of scale invariance. However, in this case the communication is performed by a single agent, and happens with a rate independent of the population size (i.e. the frequency of national emergencies does not scale linearly with population size). In these scenarios, i.e. dynamic environments, where broadcast communications happen rarely or are performed by a small pool of agents not increasing with the population size N, we can rely on the hybrid limit framework of [6], suitably extending the population semantics presented in this paper.

References

[1] Yehia Abd Alrahman, Rocco De Nicola, Michele Loreti, Francesco Tiezzi, and Roberto Vigo. A calculus for attribute-based communication. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13-17, 2015*, pages 1840–1845, 2015.

[2] Rena Bakhshi. *Gossiping Models – Formal Analysis of Epidemic Protocols*. PhD thesis, Vrije Universiteit Amsterdam, January 2011.

[3] Rena Bakhshi, Lucia Cloth, Wan Fokkink, and Boudewijn R. Haverkort. Mean-field analysis for the evaluation of gossip protocols. *SIGMETRICS Performance Evaluation Review*, 36(3):31–39, 2008.

[4] Karen Ball, Thomas G. Kurtz, Lea Popovic, and Greg Rempala. Asymptotic analysis of multiscale approximations to reaction networks. *The Annals of Applied Probability*, 16(4):1925–1961, 2006.

[5] Marco Bernardo and Roberto Gorrieri. A Tutorial on EMPA: A Theory of Concurrent Processes with Nondeterminism, Priorities, Probabilities and Time. *Theoretical Computer Science*, 202(1-2):1–54, 1998.

[6] Luca Bortolussi. Hybrid behaviour of Markov population models. *Information and Computation*, 247:37–86, April 2016.

[7] Luca Bortolussi and Nicolas Gast. Mean field approximation of uncertain stochastic models. In *Dependable Systems and Networks, DSN 2016*, pages 287–298, 2016.

[8] Luca Bortolussi and Jane Hillston. Efficient checking of individual rewards properties in markov population models. In *Quantitative Aspects of Programming Languages and Systems, QAPL 2015*, pages 32–47, 2015.

[9] Luca Bortolussi and Jane Hillston. Model checking single agent behaviours by fluid approximation. *Information and Computation*, 242:183–226, 2015.

[10] Luca Bortolussi, Jane Hillston, Diego Latella, and Mieke Massink. Continuous approximation of collective system behaviour: A tutorial. *Performance Evaluation*, 70(5):317–349, 2013.

[11] Luca Bortolussi and Roberta Lanciani. Model checking markov population models by central limit approximation. In *Quantitative Evaluation of Systems*, pages 123–138. Springer, 2013.

[12] Luca Bortolussi, Rocco De Nicola, Vasti Galpin, Stephen Gilmore, Jane Hillston, Diego Latella, Michele Loreti, and Mieke Massink. CARMA: Collective adaptive resource-sharing markovian agents. In *Proc. of the Workshop on Quantitative Analysis of Programming Languages 2015*, volume 194 of *EPTCS*, pages 16–31, 2015.

[13] Luca Bortolussi and Alberto Policriti. Stochastic concurrent constraint programming and differential equations. *Electr. Notes Theor. Comput. Sci.*, 190(3):27–42, 2007.

[14] Dario Bruneo, Marco Scarpa, Andrea Bobbio, Davide Cerotti, and Marco Gribaudo. Markovian agent modeling swarm intelligence algorithms in wireless sensor networks. *Perform. Eval.*, 69(3-4):135–149, 2012.

[15] Christopher M. Cianci, Xavier Raemy, Jim Pugh, and Alcherio Martinoli. Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics. In *Swarm Robotics, Second International Workshop, SAB 2006, Rome, Italy, September 30-October 1, 2006, Revised Selected Papers*, volume 4433 of *Lecture Notes in Computer Science*, pages 103–115. Springer, 2007.

[16] Federica Ciocchetta and Jane Hillston. Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, 410(33):3065–3084, 2009.

[17] R. W. R. Darling. Fluid Limits of Pure Jump Markov Processes: a Practical Guide. arXiv e-print math/0210109, October 2002.

[18] R.W.R. Darling and J.R. Norris. Differential equation approximations for Markov chains. *Probability Surveys*, 5:37–79, 2008.

[19] Rocco De Nicola, Diego Latella, Michele Loreti, and Mieke Massink. A uniform definition of stochastic process calculi. *ACM Comput. Surv.*, 46(1):5, 2013.

[20] Rocco De Nicola, Michele Loreti, Rosario Pugliese, and Francesco Tiezzi. A formal approach to autonomic systems programming: The SCEL language. *TAAS*, 9(2):7, 2014.

[21] B. Denby and S. Le Hègarat-Mascle. Swarm intelligence in optimisation problems. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 502(2):364 – 368, 2003. Proceedings of the VIII International Workshop on Advanced Computing and Analysis Techniques in Physics Research.

[22] Cheng Feng and Jane Hillston. PALOMA: A process algebra for located markovian agents. In *Quantitative Evaluation of Systems - 11th International Conference, QEST 2014, Florence, Italy, September 8-10, 2014. Proceedings*, volume 8657 of *LNCS*, pages 265–280. Springer, 2014.

[23] Jose Luis Fernandez-Marquez, Giovanna Di Marzo Serugendo, Sara Montagna, Mirko Viroli, and Josep Lluís Arcos. Description and composition of bio-inspired design patterns: a complete overview. *Natural Computing*, 12(1):43–67, 2013.

[24] Daniela Gavidia, Spyros Voulgaris, and Maarten Van Steen. A Gossip-based Distributed News Service for Wireless Mesh Networks. In *WONS 2006 : Third Annual Conference on Wireless On-demand Network Systems and Services*, pages 59–67, Les Ménuires (France), January 2006. INRIA, INSA Lyon, Alcatel, IFIP. http://citi.insa-lyon.fr/wons2006/index.html.

[25] J. Hillston. Fluid flow approximation of pepa models. In *2nd International Conference on the Quantitative Evaluation of Systems*, pages 33–42. IEEE, 2005.

[26] Jane Hillston. *A Compositional Approach to Performance Modelling*. CUP, 1995.

[27] Jane Hillston and Michele Loreti. CARMA eclipse plug-in: A tool supporting design and analysis of collective adaptive systems. In *Quantitative Evaluation of Systems - 13th International Conference, QEST 2016, Quebec City, QC, Canada, August 23-25, 2016, Proceedings*, volume 9826 of *Lecture Notes in Computer Science*, pages 167–171. Springer, 2016.

[28] Alireza Khadivi and Martin Hasler. Fire detection and localization using wireless sensor networks. In *Sensor Applications, Experimentation, and Logistics - First International Conference, SENSAPPEAL 2009, Athens, Greece, September 25, 2009, Revised Selected Papers*, volume 29 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 16–26. Springer, 2010.

[29] Diego Latella, Michele Loreti, and Mieke Massink. Flyfast: A mean field model checker. In Axel Legay and Tiziana Margaria, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 23rd International Conference, TACAS*, volume 10206 of *Lecture Notes in Computer Science*, pages 303–309, 2017.

[30] K.V.S. Prasad. A calculus of broadcasting systems. *Science of Computer Programming*, 25(2-3):285–327, 1995. Selected Papers of ESOP'94, the 5th European Symposium on Programming.

[31] Corrado Priami. Stochastic π-calculus. *The Computer Journal*, 38(7):578–589, 1995.

[32] David Schnoerr, Guido Sanguinetti, and Ramon Grima. Approximation and inference methods for stochastic biochemical kinetics-a tutorial review. *Journal of Physics A: Mathematical and Theoretical*, 50(9):093001, March 2017.

[33] Carolyn L. Talcott, Vivek Nigam, Farhad Arbab, and Tobias Kappé. Formal specification and analysis of robust adaptive distributed cyber-physical systems. In *Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems - 16th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2016, Bertinoro, Italy, June 20-24, 2016, Advanced Lectures*, volume 9700 of *Lecture Notes in Computer Science*, pages 1–35. Springer, 2016.

[34] Max Tschaikowski and Mirco Tribastone. Spatial fluid limits for stochastic mobile networks. *Perform. Eval.*, 109:52–76, 2017.

[35] Pavel Vrba, Vladimír Marík, Pierluigi Siano, Paulo Leitão, Gulnara Zhabelova, Valeriy Vyatkin, and Thomas I. Strasser. A review of agent and service-oriented concepts applied to intelligent energy systems. *IEEE Trans. Industrial Informatics*, 10(3):1890–1903, 2014.